DEVELOPMENT AND APPLICATION OF NOVEL COMPUTER VISION AND MACHINE LEARNING TECHNIQUES

Arthur Charles Depoian II

Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

August 2021

APPROVED:

Parthasarathy Guturu, Major Professor
Murali Varanasi, Committee Member
Kamesh Namuduri, Committee Member
Shengli Fu, Chair of the Department of Electrical Engineering
Hanchen Huang, Dean of the College of Engineering
Victor Prybutok, Dean of the Toulouse Graduate School

Depoian II, Arthur Charles. *Development and Application of Novel Computer Vision and Machine Learning Techniques*. Master of Science (Electrical Engineering), August 2021, 100 pp., 9 tables, 54 figures, 87 numbered references.

The following thesis proposes solutions to problems in two main areas of focus, computer vision and machine learning. Chapter 2 utilizes traditional computer vision methods implemented in a novel manner to successfully identify overlays contained in broadcast footage. The remaining chapters explore machine learning algorithms and apply them in various manners to big data, multi-channel image data, and ECG data. L_1 and L_2 principal component analysis (PCA) algorithms are implemented and tested against each other in Python, providing a metric for future implementations. Selected algorithms from this set are then applied in conjunction with other methods to solve three distinct problems. The first problem is that of big data error detection, where PCA is effectively paired with statistical signal processing methods to create a weighted controlled algorithm. Problem 2 is an implementation of image fusion built to detect and remove noise from multispectral satellite imagery, that performs at a high level. The final problem examines ECG medical data classification. PCA is integrated into a neural network solution that achieves a small performance degradation while requiring less then 20% of the full data size.

ACKNOWLEDGEMENTS

I need to thank a number of people starting with my Committee. Dr Guturu my advisor who took me under his wing and taught me so much. Dr Varansi who was instrumental in my continued academic pursuit, always encouraging me and guiding me, with my best interest at heart. And Dr Namuduri for being the Professor that makes us at UNT go that one step further and always believes in our potential. It has been a true honor to have them on my committee.

I would like to thank the entire school of engineering at UNT and specifically the electrical engineering department, where I studied. I want to extend a special thanks to Dr Garcia who helped me at every chance and encouraged me to press on and pursue my education.

Thank you to IEEE and HKN, the organizations and the people, for all the significant opportunities and advantages I have had over the last few years.

The team that I worked with on a daily basis in the Oscar Lab; Lorenzo Jaques, Dong Xie, Eric King, Nicholas Chiapputo, Daniel Zhang, Nathan Collins, Ethan Murrell, Hae Jin Kim, Zach Walker, Garrett Cayce, Miguel Rivera thank you for everything.

My entire family, with extra thanks to; Uncle John, Sarge, Dad Art, Dad Jack, Mom Jill, Mom Kim, Mel, Lauren and Noah. Without the encouragement and perspective of my friends Raphael Chabaneix, Jon Ashdown, Eujain Ting, and Shubham Chamadia, I couldn't have gotten here. Finally, the most important of all my wife, Colleen, Thank You!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PREVIOUSLY PUBLISHED WORKS PRODUCED WITH PERMISSION

This thesis is based in part on the previously published articles listed below. All articles contained within the thesis are reproduced with permission from the International Society for Optics and Photonics and my co-authors.

(1) Chapter 1: Arthur C. Depoian II, Lorenzo E Jaques, Dong Xie, Colleen P. Bailey, and Parthasarathy Guturu, *Computer vision learning techniques for sports video analytics: removing overlays*, Big Data II: Learning, Analytics, and Applications, vol. 11395, International Society for Optics and Photonics, SPIE, 2020, pp. 61–72; https://doi.org/10.1117/12.2560888

(2) Chapter 3: Arthur C. Depoian II, Eric King, Colleen P. Bailey and Parthasarathy Guturu, *Meteorological data outlier detection: a principal component approach*, Remote Sensing for Agriculture, Ecosystems, and Hydrology XXII, vol. 11528, International Society for Optics and Photonics, SPIE, 2020, pp. 159–164; https://doi.org/10.1117/12.2574178

(3) Chapter 4: Eric King, Arthur C. Depoian II, Colleen P. Bailey and Parthasarathy Guturu, *Weighted principal component analysis fusion of satellite telemetry data*, Remote Sensing of the Ocean, Sea Ice, Coastal Waters, and Large Water Regions 2020, vol. 11529, International Society for Optics and Photonics, SPIE, 2020, pp. 9–15; https://doi.org/10.1117/12.2574183

(4) Chapter 4: Arthur C. Depoian II, Lorenzo E. Jaques, Dong Xie, Colleen P. Bailey, and Parthasarathy Guturu, *Neural network image fusion with PCA preprocessing*, Big Data III: Learning, Analytics, and Applications, vol. 11730, International Society for Optics and Photonics, SPIE, 2021, pp. 132–147; https://doi.org/10.1117/12.2588039

(5) Chapter 5: Lorenzo E. Jaques, Arthur C Depoian II, Dong Xie, Colleen P. Bailey, and Parthasarathy Guturu, *A machine learning approach to medical data identi-*

fication through principal component analysis, Big Data III: Learning, Analytics, and Applications, vol. 11730, International Society for Optics and Photonics, SPIE, 2021, pp. 7–15; https://doi.org/10.1117/12.2586038

CHAPTER 1

INTRODUCTION

This thesis is the culmination of 2 years of graduate work in the field on signal processing, with concentrations into machine learning and computer vision, using multiple techniques. The first chapter contains a novel overlay identification procedure that was designed using mean window filter masking to process large amounts of video footage. The following chapter is the development and profiling of principal component analysis (PCA) functions that solve for the L_1 and L_2 principal components, one of the most prolifically used techniques in signal processing. The next chapter applies the knowledge of the previous chapter paired with a new thresholding algorithm to big weather data to remove errors from the dataset. This than flows into the application of a novel controlled image fusion algorithm that uses PCA to identify the similarity of the data contained in multi-spectral satellite images for fusion. This work led to a further investigation into the application of a neural network to denoise the samples deemed inferior. The final chapter is a prototype for the application of PCA dimensionality reduction after manipulation of the original data to medical data classification neural networks, leading to over 80% compression at the input to the neural network.

Big data has been driving professional sports over the last decade. In our data-driven world, it becomes important to find additional methods for the analysis of both games and athletes. There is an abundance of videos taken in professional and amateur sports. Player, team, and sport specific datasets can be created utilizing computer vision techniques. The novel approach presented in this thesis creates an autonomous masking algorithm that can receive live or previously recorded video footage of sporting events and identify graphical overlays. This method can be used to optimize further processing including tracking and text recognition algorithms for real-time analysis.

PCA, when applied to electrical engineering problems, has proven itself to be a valuable tool at every turn. With wide use in information theory, control systems, computational

networks, communication networks, and machine learning, it has helped identify or solve the optimal solutions. For over 100 years the options to calculate principle components using of the L_2 variety has steadily evolved into a reliable function set. Recently, the L_1 optimal exact solution was found, advancing the field greatly. The superiority of L_1 PCA is vast when compared to that of L_2 in applications where outlier rejection is required. The newest versions of the L_1 functions advance it further, swapping out the exhaustive search procedure for the well know advantages of single bit flipping techniques. The options are profiled against each other and developed for the Python language, taking advantage of the multiple libraries and exploration of rapid prototyping to the graphics card as a computational device. Analysis of L_2 solving techniques is also documented for computation time.

Satellites are equipped with an array of diversified sensors, capable of relaying multiple types of optical data about the earth's surface. The different sensors can capture varying levels of detail for a particular area of interest. Combining information gathered from sensors, ranging from the infrared to the visible spectrum, can enhance visualization and depth of data. Denoiseing of images with autoencoders has shown great promise over the last few years, with results superior to that of classic signal processing. The application of principal component analysis to data fusion is traditionally processed by weighted reliability matrix methods. A novel weighted reliability with rejection controlled PCA based sensor algorithm is developed to improve data fusion quality, creating a more robust visualization of the composite information obtained from satellites. The developed algorithms can be applied using both L_2 and L_1 PCA. Simulation studies validate the proposed controlled weighted fusion method, even under high levels of corruption. These advances are then paired with a autoencoder trained to denoise images, with the goal of retaining even more of the sensor information.

Meteorological modeling takes data captured from multiple sources that is then processed by data mining techniques to predict environmental changes. The most commonly used machine learning techniques for processing meteorological data are decision trees, rule-based methods, neural networks, naive Bayes, Bayesian belief networks, and support vector

machines. These techniques require accurate data for effective models to be simulated. Meteorological datasets can contain outliers and errors that can significantly skew the accuracy of the generated models that are relied upon for many sectors of society including agriculture, natural disasters, and meteorological forecasting. The method presented eliminates outliers from meteorological data to enhance the accuracy of models by applying a blind thresholding algorithm to the principal components (PCs) obtained from L_1 and L_2 norm principal component analysis to identify and discard outliers in the dataset.

Recent years have seen the growth of big data into all fields, with great interest in the expansion of data available for analysis in medicine. The application of advanced signal processing techniques and algorithms to the medical sector has shown great promise for detection and confirmation of symptoms in patients. As the information size has increased to the level of big data, more advanced machine learning solutions are applied to help determine irregularities in the data and train the system to focus on specific locations. The implementation of dimensionality reduction to clarify the features for extraction can lead to more efficient analysis at a small accuracy cost. The algorithm presented utilizes a convolutional neural network to categorize electrocardiogram (ECG) data by processing the original data, implementing the fast Fourier transform (FFT) and principal component analysis (PCA) to reduce dimensionality, while maintaining a considerable level of performance. The three intelligent identification algorithms are produced as a modular piece that can be fed into other specialized machine learning systems or analyzed using traditional diagnostic procedures.

CHAPTER 2

COMPUTER VISION: OVERLAY IDENTIFICATION

Object detection is one of the classical objectives of the computer vision field. Video processing methods have achieved great results in recent history, typically relying on the frame being raw with no modifications [74]. The classic application of video object detection to the problems of traffic control, intelligent video surveillance and robot navigation have evolved greatly recently [28, 31, 57], becoming a part of our daily life. When these same methods are exposed to video that has been post processed, for instance the broadcast of sports events, it is commonplace to add graphical overlays. Overlays add additional information to the broadcast for the viewer, at a cost to the pure application of classic techniques to the frame. One issue created is the addition of superior edges that will wreak havoc on most tracking algorithms. Like many other computer vision problems, object detection has found solutions to the problems of motion blur, virtual focus, target motion, occlusion, and low/inconsistent video quality [63, 86]. Key features or objects are traditionally identified using histogram of oriented gradient (HOG) [14], scale-invariant feature transform (SIFT) [15, 43], and/or the deformable part model (DPM) [19, 82]. Other target detection methods including selective search and edge boxes [77, 87] use a candidate window, depending on the application. The use of multiple cameras and fixed positioning are the currently used techniques of current research, adding valuable information to the sports analytics database [56]. The access to this game footage is costly and sometimes unobtainable, creating a barrier to wide spread application of new techniques. The techniques and processes presented grew from initial exploration of viable camera and object motion analysis, and the identification of a need that arose early on. Data that is openly available, when optical flow or frame-to-frame transform algorithms are applied in conjunction or alone, leads to results that are corrupted or of limited accuracy. The application of advanced cropping leads to more accurate data tracking of the actions taking place during the event. A novel solution is created to identify and crop these overlay graphics from the frame when detected, seeking to preserve

the maximum image quality of the original frame. The method is prototyped in the Python environment and tested on HD video recordings from multiple live broadcasts of basketball in both 720p and 1080p format. The software autonomously trains across a time window, building its database of proposed masks. The frames are intelligently cropped, preparing them for processing by classic computer vision algorithms. The technique proposed by Ekin and Jasinschi [16] process graphical overlays for removal using a distinctly different method. Detection and estimation of the occurrence of overlays is handled by statistical learning in contrast to their approach of color detection.

2.1. Background

This section introduces some of the computer vision methods utilized in the proposed graphical overlay identification and removal method.

2.1.1. Otsu's Method

The act of thresholding can be viewed as a global or local operation that can be constant or dynamic. The optimal binary threshold is usually determined using Otsu's method [60], a dynamic threshold that is traditionally applied on a global scale. The process separates a range of values into two classes, usually images; the foreground class and the background class [84]. The threshold algorithm seeks to maximize the separation of the two identified classes, ensuring the optimal separation from one another. It establishes the new distributions with the least class variance and the greatest difference in class means. The resultant is tightly clustered sets of data. In computer vision applications, the pixel value conditional probabilities are calculated to distribute into each of the binary classes. The probability distribution p_i (1) is the normalization of a single channel image.

$$p_i = \frac{number\ of\ pixels\ at\ intensity}{number\ of\ pixels} = \frac{n_i}{N} \tag{1}$$

$$p_i \geq 0,\ \sum_{i=0}^{L} p_i = 1 \tag{2}$$

The two probabilities of class occurrence are:

$$w_0 = Pr(C_0) = \sum_{i=0}^{k} p_i = \omega(k) \tag{3}$$

$$w_1 = Pr(C_1) = \sum_{i=k+1}^{L} p_i = 1 - \omega(k) \tag{4}$$

These probabilities can then be used to find the respective intensity mean value vectors, as well as the total mean vector.

$$\mu_0 = \sum_{i=1}^{k} i \ Pr(i|C_0) = \sum_{i=1}^{k} i\frac{p_i}{\omega_0} = \frac{\mu(k)}{\omega(k)} \tag{5}$$

$$\mu_1 = \sum_{i=k+1}^{L} i \ Pr(i|C_1) = \sum_{i=k+1}^{L} i\frac{p_i}{\omega_1} = \frac{\mu_T - \mu(k)}{1 - \omega(k)} \tag{6}$$

where

$$\omega(k) = \sum_{i=1}^{k} p_i \tag{7}$$

$$\mu(k) = \sum_{i=1}^{k} i \ p_i \tag{8}$$

The total mean level of the original image is:

$$\mu_T = \mu(L) = \sum_{i=1}^{L} i \ p_i \tag{9}$$

Solving for the class variances σ_0^2 and σ_1^2:

$$\sigma_0^2 = \sum_{i=1}^{k} (i - \mu_0)^2 \ Pr(i|C_0) = \sum_{i=1}^{k} (i - \mu_0)^2 \ \frac{p_i}{\omega_0} \tag{10}$$

$$\sigma_1^2 = \sum_{i=k+1}^{L} (i - \mu_1)^2 \ Pr(i|C_1) = \sum_{i=k+1}^{L} (i - \mu_1)^2 \ \frac{p_i}{\omega_1} \tag{11}$$

$$\sigma_T^2 = \sum_{i=1}^{L} (i - \mu_T)^2 p_i \tag{12}$$

The global variance is the squared distance from the global mean as displayed in

Equation (12). This shows how far apart the histogram of the image is, where the worst case is uniform and the best case is a delta function. Otsu can fail when there are no strong peaks in the histogram. For example, this occurs when the object is small with respect to the background. This can be achieved by implementing a low pass filter or by only considering pixels near the edges when computing the threshold by drawing a boundary of the interior and exterior edges. Another option is to calculate variable adaptive thresholding where the frame is broken up into blocks and Otsu thresholding per block is applied to remedy the precision.

2.1.2. Sobel Operator

Kernel convolution for Sobel edge detection identifies the region of an image that has a sharp change in intensity or a sharp change in color as shown in Equation (13). A high value indicates a steep change and a low value indicates a shallow change [10, 75, 25, 52]. The Sobel operation is the approximation of the derivatives of an image for the x and y direction separately [73, 38]. The image is converted from color to grayscale when a Gaussian Blur is applied. After the preprocessing is completed, the Sobel operators are then utilized to check the change in intensity. This Gaussian Blur is modified in accordance with the desired output to increase/decrease the amount of noise left in the picture.

$$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * A, \qquad G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & 0 & -1 \end{bmatrix} * A \tag{13}$$

G_x finds how strong the gradient is on the x-axis and G_y finds the strength of the gradient on the y-axis. When totaled, the general magnitude is found, as shown in Equation (14).

$$G = \sqrt{G_x^2 + G_y^2} \tag{14}$$

$$\Theta = atan(\frac{G_y}{G_x}) \tag{15}$$

If G_x or G_y is large and the other is small, then a moderately large gradient is achieved. If the gradient in both directions is large, then a very large gradient is obtained. These values define how big the edge is at the specified location.

2.1.3. Canny Edge

In its simplest form, the input of the Canny Edge detector is the output of a Sobel operator that is processed in the x and y directions in which the gradient is calculated. The orientation of those gradient values is the starting point to begin the Canny Edge detection process. Canny Edge detection operates by taking the image from the Sobel output and thinning all the edges to a width of one pixel. The operation is designed to focus on where the edges are located and not their thickness. If there is a high-resolution image, then the gradient is spread over many pixels. In a low-resolution image, the same gradients will appear sharper.

Canny Edge first finds the edges, then uses the hysteresis thresholding process, which is also a two-level threshold. This process determines every pixel and tries to find the local maximum of clusters as the edge is approached. This means the pixel's value is larger than its neighbors, retaining the orientation that was produced by the Sobel at every occurrence.

The value of the edge might be smaller or larger based on what is necessary to determine whether x is larger than its neighbors across the edge. Canny Edge checks the orientation of the edge based on the output of the Sobel operator. The tangent inverse is taken and becomes the new local maxima if the value is larger than its neighbors. This is completed in order to produce thin edges at the peaks of the center response. If there is a gradient on the edge of an object in the frame, the operation finds the strongest x value of the neighbor pixel cluster. The Canny Edge finds the centroid of the pixel cluster and selects that pixel as the strongest value.

The next step is to remove the edges that have a weak response as they are likely to be noise. This is done to create an image of dominant edges and preserve only those where hysteresis thresholding occurs [10, 80]. This determines which edges are important and which are not. All edges are graded from 0 to 255, where 255 is the strongest edge

possible. For example, a value of 15 will show almost all edges, equating to a low signal to noise ratio. Conversely, a value at 240 creates a hard outline of the overall object, in which some of the less dense edges will be lost. The hysteresis thresholding preserves the core edges for the overall image layout.

2.1.4. Morphology

Morphology is the application of a structured element to a binary image, usually after thresholding. The element of action is created using predefined functions of shapes dependent on the desired operation. The most commonly used morphological structures are that of erosion, dilation, disk and box. Examples of each method can be seen in Figure 2.1.

Erosion removes pixels on object boundaries which leads to eliminating noise at the edge boundaries and decreases the size of the object. Dilation adds pixels to the boundaries of objects in an image through a process that fills in holes and increases the size of the object. Depending on the combinations of erosion, or dilation, the output can either be the 'opening' or 'closing' of the image.

$$A \ominus B = \{z | B_z \subseteq A\} \tag{16}$$

Erosion (Figure 2.1.A) is defined as the set of z such that the structuring element translated by z fits fully inside A, Equation (16). It helps negate extra artifacts outside of the centroid image and can also help break isthmus connecting pixels. Since more desired structuring elements are required depending on the orientation, the structuring element should be chosen

1	1	0	1	1
1	1	0	1	1
0	0	0	0	0
1	1	0	1	1
1	1	0	1	1

A. Erosion

0	0	1	0	0
0	0	1	0	0
1	1	1	1	1
0	0	1	0	0
0	0	1	0	0

B. Dilation

0	1	1	1	0
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
0	1	1	1	0

C. Disk

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

D. Box

FIGURE 2.1. Morphology Structure Elements

accordingly. Erosion removes the thin lines and isolated dots while leaving the gross detail. The larger the structuring element, the more points are removed from the binary image, which are also referred to as layers. Dilation, Equation (17), is used to bridge gaps that are smaller than the structuring element. This is useful when there is too low a threshold to fill in objects, as shown in Figure 2.1.B.

$$A \oplus B = \{z | \hat{B}_z \cap A \subseteq A\} \tag{17}$$

Opening and Closing operators are applied to bridge gaps/fill holes, but do not change the overall size of the object, meaning that it neither thins out or fattens an image to keep it looking roughly the same while fixing little problems. Opening is the process used to erode and then dilate to break narrow bridges and eliminate thin structures, its formulation is $A \circ B = (A \ominus B) \oplus B$. Closing is designed to eliminate small holes through the dilation and then erosion fusing narrow breaks. It can be symbolized as $A \cdot B = (A \oplus B) \ominus B$.

2.2. Implementation

The algorithm design presented in Figure 2.2 takes in a set of approximately 2000 consecutive frames or one minute of game footage at thirty frames per second for initial training. It then dynamically crops graphical overlay through the total system integration of multiple signal and image processing techniques.

The frames are read into a buffer and grayscaled, isolating the luminance intensity of each pixel. Preprocessing of each frame utilizing Otsu's method to threshold these values helps to even out frame-to-frame intensity and prepares it for Canny Edge detection. Application of the classic Canny Edge algorithm produces a result of superior accuracy. After this, the frame of edges is accumulated over a buffering window dependent on the frames of the original source. The window of edges is then mean filtered [69], which results in a raw training frame. This frame is then blurred to connect the filtered edges. Thresholding is carried out again to generate the highs and lows to increase value separation. A size dependent disk blur is then applied to the frame. To further increase the chance of detection, the frame is eroded and then dilated. At this point, the frame is significantly altered and ready

for computational analysis.

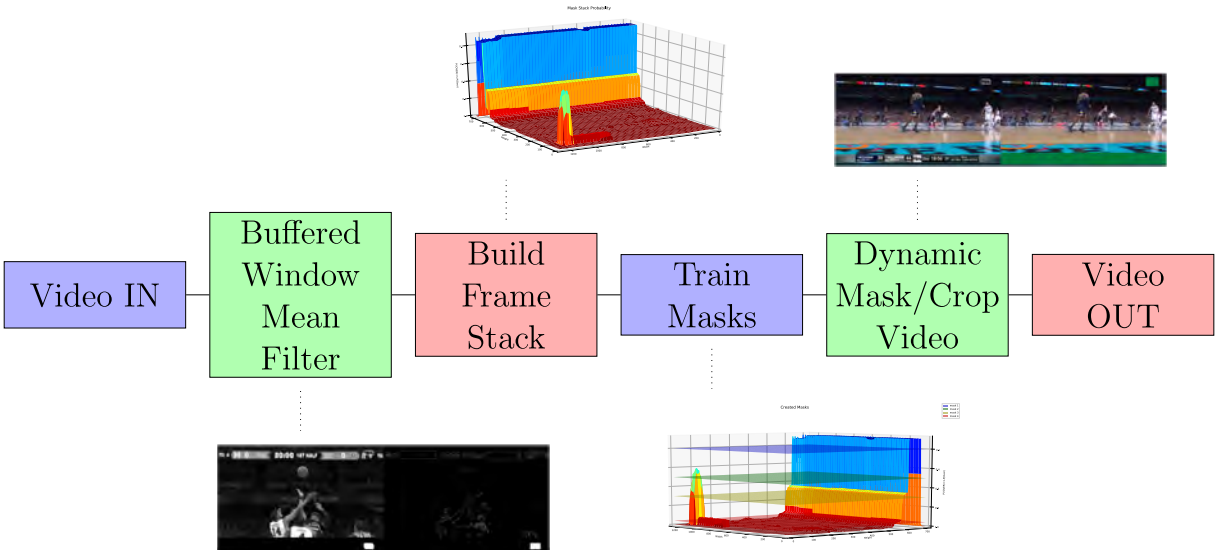


FIGURE 2.2. Algorithm Flow Chart

The resulting frame is passed to a function that identifies key contours and logs the location while seeking to interconnect these vectors into known shapes. After identification, the training mask is created on a blank frame of equal size to the original frame, utilizing contour-drawing functions packaged with Open CV. This process is detailed in Algorithm 2.

The training-mask frames are accumulated for further processing. Upon completion of training, the accumulated training mask frames are stored and used for further implementation. The mask dataset is created through the further use of Otsu's algorithm. The stack of frames is looped until Otsu's method fails to produce values over 50% of the range. During the loop, intensity contours are identified and converted to anchored shapes. Those shapes are applied to blank frames that are subtracted from the stack before returning for further loop evaluation. The masks are then compared to each other and the original frame size. Those that do not have enough of an effect when applied are removed.

The mask database is then stored as a trained dataset for future use. The active overlay is achieved through the use of a very similar filtering technique with a smaller buffer and looser processing, allowing for real-time application of the masking database. The masks

are boolean type arrays, allowing for quick processing against the altered image. Binary operations apply the mask to the filtered frame and calculate the relative coverage of intense pixels. The decision is made dynamically by taking into account the probability of the originally created mask in the database. The resulting masks are applied to the frame and either displayed or stored. The overall system is described in Algorithm 1 which utilizes the previously described Algorithm 2 to both train and automatically crop.

Algorithm 1 MAIN Active Video Crop

Inputs:
    $Load(TrainingVideo, WholeVideo$
$ProbabilityStack \leftarrow CannyMeanFilter(TrainingVideo)$
$Masks \leftarrow MaskCreator(ProbabilityStack)$
$MaskedVideo, CroppedVideo \leftarrow MaskPlayer(WholeVideo, Masks)$

Algorithm 2 Canny Mean Filter

while Video frame available **do**
    $gray \leftarrow frame$
    $threshold \leftarrow Otsu(gray)$
    $edge \leftarrow Canny(threshold)$
    $Roll\ Edge$
    $Edge[0] \leftarrow edge$
    $Average \leftarrow mean(Edge[0] : Edge[window])$
    for i **do**
        if $Average[i]/max(Average) < 0.85$ **then**
            $Average[i] \leftarrow 0$
        end if
    end for
    $blur \leftarrow SquareBlur(Average)$
    $binarythreshold \leftarrow Binary(Otsu(blur))$
    $morph \leftarrow Morphography(binarythreshold)$
    $box \leftarrow FindCurves(morph)$
    $Meanshapes \leftarrow DrawCurves(box)$
end while

2.3. Results

The NCAA 2018 National Championship [45] and NCAA 2019 National Championship [46] basketball games were used as the video dataset to test the proposed algorithm. The preprocessing is shown in Figure 2.3, starting with the frame being grayscaled and the

application of Otsu's method in the upper-left quadrant. The applied Canny Edge detection is shown in the upper-right quadrant on the same frame of game action. The morphed training edges are shown in the lower-left quadrant after mean window filtering is applied. The lower-right quadrant output is the result of the connected polygon drawing, estimated from the average edges based on the location of pixel intensity vectors. This creates the hard edge of a near-perfect rectangle. This preprocessing is shown in Figure 2.4 and Figure 2.5 over the duration of a game.

FIGURE 2.3. Game Tip-Off - Gray Scaled & Otsu (upper-left), Canny Edge (upper-right), Combination of Erosion/Dilation (lower-left), Average Edge (lower-right)

Figure 2.6 displays the probability of occurrence of each pixel given the mean filtering where the x-axis and y-axis equal the frame dimensions, 1280 by 720 respectively. This is generated from training data of one minute of game-time (1.3% of the total footage) or approximately 1900 frames at 30 frames per second.

Figure 2.7 provides the thresholds of probability where each mask is applied to the frame overlayed on the intensity of the probability stack. Mask 1 (blue transparent rhombus), i.e. the scoreboard at the bottom of the basketball frame, ended up at 20 percent. This means that approximately 80 percent of the frames were lacking motion in that pixel mask cluster. Mask 2 (green transparent rhombus), i.e. the TBS logo, achieved roughly 50 percent

FIGURE 2.4. Traditional Right-Court View - Gray Scaled & Otsu (upper-left), Canny Edge (upper-right), Combination of Erosion/Dilation (lower-left), Average Edge (lower-right)

FIGURE 2.5. Scene Change - Gray Scaled & Otsu (upper-left), Canny Edge (upper-right), Combination of Erosion/Dilation (lower-left), Average Edge (lower-right)

lack of motion. This was due to the opaqueness of the logo as well as the pixels blending with background through an audience members' lighter color clothing or the varying white spaces of the basketball arena. Mask 3 (yellow transparent rhombus), equating to the status ticker update, was around 70 percent, meaning that for 30 percent of the frames there was

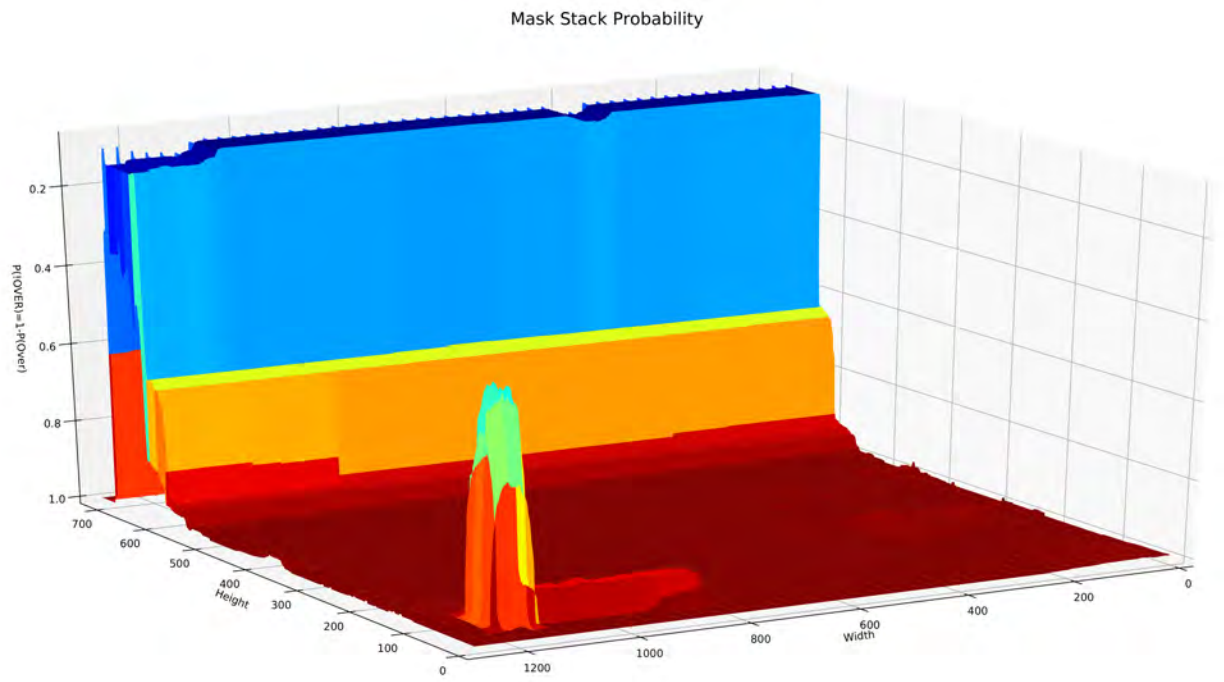


FIGURE 2.6. Mask Stack Probability for Approximately 1900 Frames

a featured update for when a team is on a scoring run or when an individual stat line was displayed. An example of a featured update is shown in Figure 2.10. Mask 4 (red transparent rhombus) was at roughly 90 percent, meaning that around 10 percent of the frames had the 'Teamcast' overlay as shown in Figure 2.4.

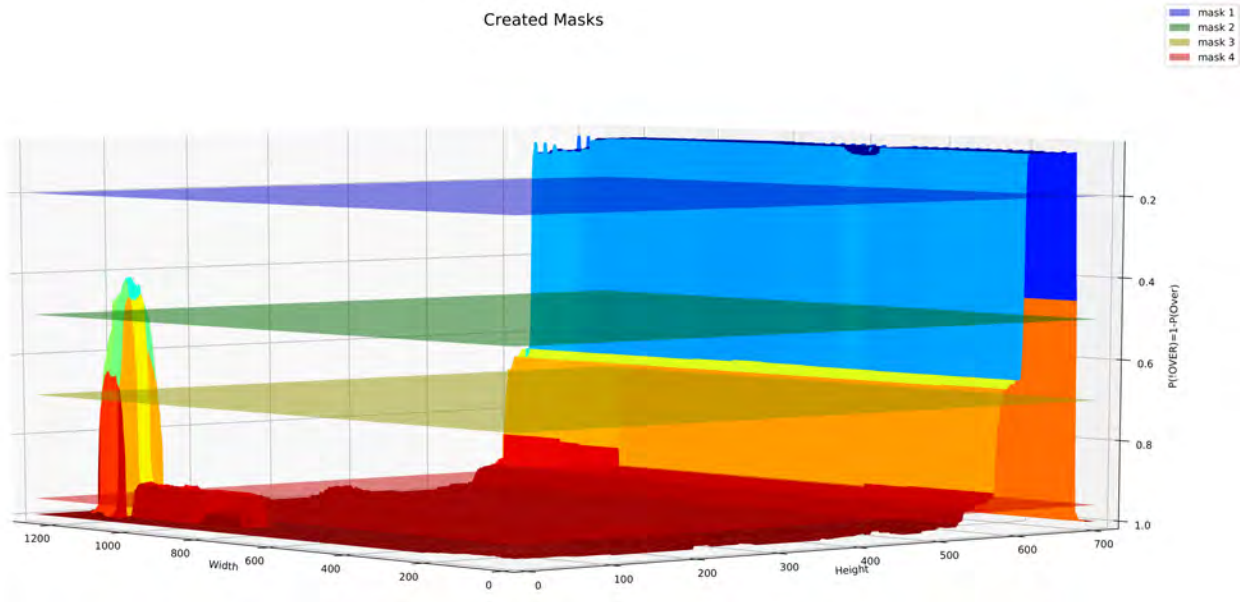


FIGURE 2.7. Created Masks for Approximately 1900 Frames

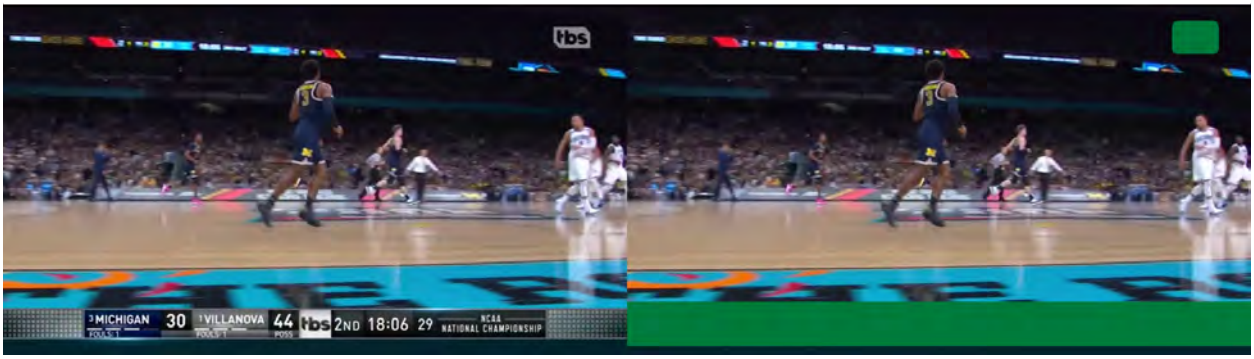


FIGURE 2.8. Side-Court View - Raw frame footage (left), Algorithm output with dynamic masks 1 & 2 applied (right)

FIGURE 2.9. Skycam-View of fans - Raw frame footage (left), Algorithm output with dynamic masks 1, 2, & 3 applied (right)

FIGURE 2.10. Close-Up View - Raw frame footage (left), Algorithm output with dynamic masks 1, 2, & 3 applied (right)

Samples of the working algorithm are shown in Figure 2.8, Figure 2.9, Figure 2.10, Figure 2.11 & Figure 2.12. Further examination of Figure 2.12 shows that the algorithm does not react to the TBS logo during a shot transition splash graphic that is not part of an overlay.

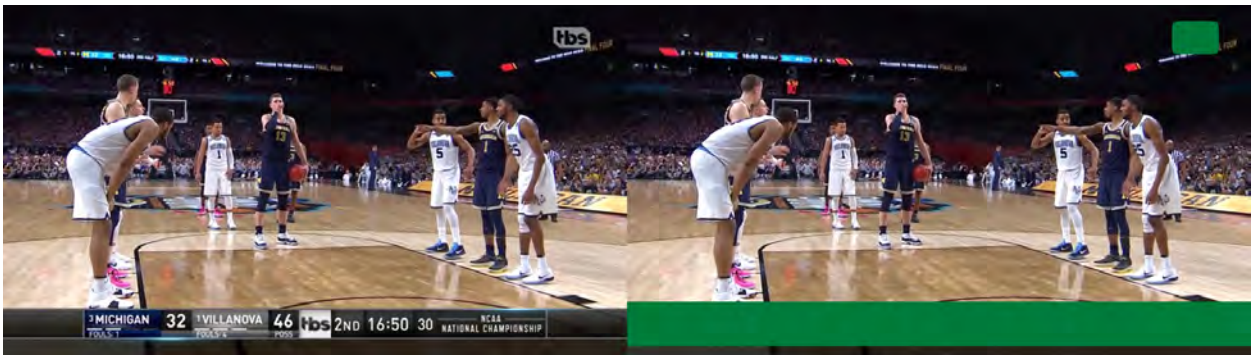


FIGURE 2.11. Free-Throw View - Raw frame footage (left), Algorithm output with dynamic masks 1 & 2 applied (right)

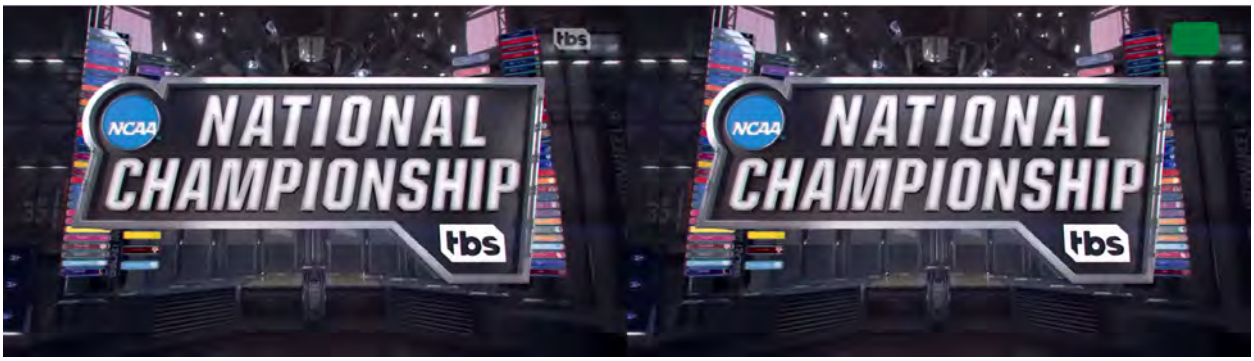


FIGURE 2.12. NCAA TV Scene Change - Raw frame footage (left), Algorithm output with dynamic mask 1 applied (right)

2.4. Conclusion

An unsupervised algorithm to effectively crop graphical overlays applied to broadcast sporting events is shown to produce viable results. The development of masks from a relatively small training window produces a set that properly identifies overlays with no human pre-processing. The experimental studies shown in the previous section, present evidence that the proposed method can detect the overlayed graphics onto the frame in different lighting and color variance conditions. It can then be deduced that this method has the effectiveness and practicality to aid in many computer vision applications.

Further research into this novel approach to autonomous overlay cropping includes expanding the database of trained sets and improving the mask decision algorithm. Including smarter tools such as text and object recognition to identify network, sport, and team could be of great value. The alternative profitability of this algorithm is to apply market-

ing materials to live broadcasts or previous recordings to target specific audiences and not occlude the game.

CHAPTER 3

L_1 & L_2 PRINCIPAL-COMPONENT ANALYSIS (PCA) COMPUTATION TECHNIQUES

PCA is traditionally applied to a matrix as a tool for dimensionality reduction, seeking to symbolize the variance of the data in a form that preserves as much information as possible in a reduced space. Over the past decades principal component analysis has been applied to many fields and comes in many forms including Karhunen-Loève transformation, the Hotelling transformation, the method of empirical orthogonal functions, and singular value decomposition. In more recent history advances have been made into an optimal solution for L_1 principal components [47]. Depending on the style of PCA implemented and the number of components preserved for each distinct matrix, the resultant can be considered reversible or non-reversible. This same matrix may still contain enough information to infer the original matrix characteristics and contain valuable information to solve real world problems [39, 2, 55].

In *On lines and planes of closest fit to systems of points in space* [20], Pearson illustrates the basic idea of dimensionality reduction as a technique to preserve the characteristics of the data based upon mean, deviation, and variance. Further analysis into the value of dimensionality reduction and its challenges are well documented by Bellman in *Dynamic Programming* [5]. He coins the 'Curse of Dimensionality' term and develops the mathematical basis for his reasoning, presenting it in terms of computational cost created by the addition of a new dimension leading to an exponential growth in cost. The works of Bellman et. Dreyfus [6] were investigated and reasoned through as a basis for inferences as the development of the connection between the mathematical with that of the computational. The many examples were invaluable assets to the study of the evolution of the PCA algorithms through the years. In literature, it is a consensus that the use of Single Value Decomposition (SVD) is the technique to employ to find the L_2 Principle Components for its computational speed and simplicity. A technique proposed by Golub and Kahan in 1965 [21], which uti-

lizes Householder transformations greatly advanced the ability to efficiently solve the SVD. The algorithm was further advanced by Golub and Reinsch [22] in 1970, creating the most popular method applied today to solve for the SVD of a given matrix. The algorithms used have evolved over the years further only slightly from this one to take advantage of the computational architecture of the target device. A further analysis is provided by Karystinos in *Optimal Algorithms for Binary, Sparse, and L_1-Norm Principal Component Analysis* [30], detailing each step and additional applications. The optimal solution utilizes an exhaustive search method paired with SVD for the solution. A computational advancement of this algorithm to cut the computational cost was found by Markoupoulous et al. [48], in which Single Bit Flipping (SBF) is used to find the principal components at a considerable savings when compared to the exhaustive method. SBF has shown itself many times over to significantly improve the performance of many algorithms that relied previously upon exhaustive search [17]. There are two methods considered, the first one is that of Kundu et al. where they develop the fast method using SBF and no SVD, using an iterative method to compute each set of PCs in series [35], in this study we will refer to it as the Fast method. The second method used the SVD, then randomly seeds the proposed binary vector using SBF methods, this method can be found in publicly shared code repositories based upon [49].

With the ever increasing computational power at hand in current computers, there is a need to explore the different ways to leverage all this power. With many new applications having machine learning principles applied everyday, the use of PCA as a solution and as a part of the total solution with grow too. These studies will investigate the performance of the most readily available SVD algorithms in the Python environment to solve the L_2 norms and implement multiple versions of the SBF algorithms to solve L_1 norms.

3.1. Methodology

For a data matrix $\mathbf{X} \in \mathbb{R}^{D\times N}$, the L_1 and L_2 principal components $\mathbf{Q}$ can be found by solving the corresponding equations.

$$L_1 : \quad \underset{\mathbf{Q}\in\mathbb{R}^{D\times K},\, \mathbf{Q}^T\mathbf{Q}=\mathbf{I}_K}{\arg\max} \|\mathbf{X}^T\mathbf{Q}\|_1 \tag{18}$$

$$L_2 : \quad \underset{\mathbf{Q}\in\mathbb{R}^{D\times K},\, \mathbf{Q}^T\mathbf{Q}=\mathbf{I}_K}{\arg\max} \|\mathbf{X}^T\mathbf{Q}\|_2 \tag{19}$$

3.1.1. Algorithms

The L_2 PCs are calculated via the SVD method based upon Algorithm 3. There are a number of SVD functions available for implementation in the Python language. This study will concentrate on the functions contained in SCIPY [79], NUMPY [23], CUPY [58] and a experimental SKlearn [68] randomized SVD function. Most of these functions utilize the CPU to calculate the SVD except CUPY which runs on the GPU, specifically recent Nvidia GPUs. A Python library for finding the L_1 PCs using the single bit flipping method that relies on the calculation of the SVD of a matrix for its solution is explored in Algorithm 4. The SVD based method is also modified by swapping out the SCIPY SVD for that of NUMPY and the SKlearn's random SVD functions. The Fast Algorithm 6 is also built two ways and explored in Python using the NUMPY library and the powerful PyTorch library [66] to execute the solution on the GPU or CPU.

Algorithm 3 L2 Projection and PCs using SVD

Inputs:
$X_{M\times N}, Rank$
$U, \Sigma, V \leftarrow svd(X)$
$U_{PC} \leftarrow U[:, : Rank]$
$V_{PC} \leftarrow V[: Rank, :]$
$\Sigma_{PC} \leftarrow \Sigma[: Rank]$
$Q_{PC} \leftarrow U_{PC} * V_{PC}$
Output:
$Q_{PC}, U_{PC}, \Sigma_{PC}, V_{PC}$

3.2. Testing

The algorithms are exposed to identical information at each testing iteration and run on a series of current machines available, as specified in Table 3.1. A series of testing was created to profile the SVD algorithms against each other while keeping track of the time required to find the solution. To profile the L_1 solving algorithms, the L_1 SBF library and

Algorithm 4 $L_1bf\_svd\_K$

Inputs:
$X_{M\times N}, rank = K$
$b_{bf} \leftarrow sbf\_K(X, K)$
$U, \Sigma, V \leftarrow svd(X \times b_bf)$
$U_{PC} \leftarrow U[:, : Rank]$
$V_{PC} \leftarrow V[: Rank, :]$
$Q_{PC} \leftarrow U_{PC} * V_{PC}$
Output:
Q_{bf}

Algorithm 5 sbf_K

Inputs:
A, K
$U, \Sigma, V \leftarrow svd(A)$
$Y \leftarrow \Sigma V^T$
$b \leftarrow sgn(V[:, 0]$
$nn = nuclear\_norm(Y \times b)$
for $i < K$ **do**
    $\alpha \leftarrow (\Sigma_n nuclear\_norm(Y \times b[:, i](-@n) \forall n \in \{1, ..., N\}$
    if $max(\alpha) > nn$ **then**
        $nn = max(\alpha)$
        $b[argmax(\alpha), i] \leftarrow -b[argmax(\alpha), i]$
    end if
end for
$B \leftarrow b[argmax(nn), :]$
Output:
B

Algorithm 6 $L_1 : Fast\_sbf\_K$

Inputs:
$X_{M\times N}, K$
$\hat{X} = X$
for $i < K$ **do**
    $Q_p[:, i] \leftarrow Fast\_sbf\_one(\hat{X})$
    $\hat{X} \leftarrow \hat{X} - ((Q_p[:, i] \times Q_p[:, i]^T) \times \hat{X})$
end for
Output:
Q_p

the newly created functions, testing was carried out in much the same way, exposing each function to the identical matrix and timing the process with the addition of a random rank variable as there are many applications when less then full rank is valuable, the PyTorch

Algorithm 7 $L_1 : Fast\_sbf\_one$

Inputs:
$X_{M\times N}$
$[B]_{:,n} \leftarrow sbf\_fast(X, sgn([X^T X]_{:,n}))\forall n$
$c \leftarrow argmax_i [B^T X^T X B]_{i,i}$
Output:
$\hat{r}_{L_1} \leftarrow X[B]_{:,c}/||X[B]_{:,c}||_2$

Algorithm 8 sbf_fast

Inputs:
$X_{M\times N}, b \in \{\pm 1\}^N$
while True **do**
    $\alpha_i \leftarrow b_i(\Sigma_{j\neq i}[B]_{:,n} b_j [X^T X]_{i,j} \forall i \in \{1, ..., N\}$
    $(v, p) \leftarrow minimum(\alpha)$
    if $v < 0$ **then**
        $b_p \leftarrow -b_p$
    else EXIT
    end if
end while
Output:
b_p

TABLE 3.1. Devices used for Testing of the Algorithms

Device	CPU (Intel)	RAM (DDR4-3200)	GPU (Nvidia)
1	I9-9900k	64GB	2070 Super
2	I7-9700k	32GB	2070 Super
3	I7-9700k	16GB	1660 Super
4	I5-9600K	32GB	1660 Super
5	I5-9600K	16GB	1660 Super

code is executed on the GPU.

3.3. Results

The results of 1035 synchronous experiments of the L_1 algorithm testing procedure are documented in table 3.2 where it becomes apparent from the numbers that the Fast algorithm is speedy at calculating the PCs per Rank. There is a good argument made to implement this function in solving low rank PCs of large matrices and equally an argument as to the implementation of the SVD SBF method when looking for full or close to full rank PCs. The plot in Figure 3.1 shows the curves of each algorithm on a log scale, shedding further

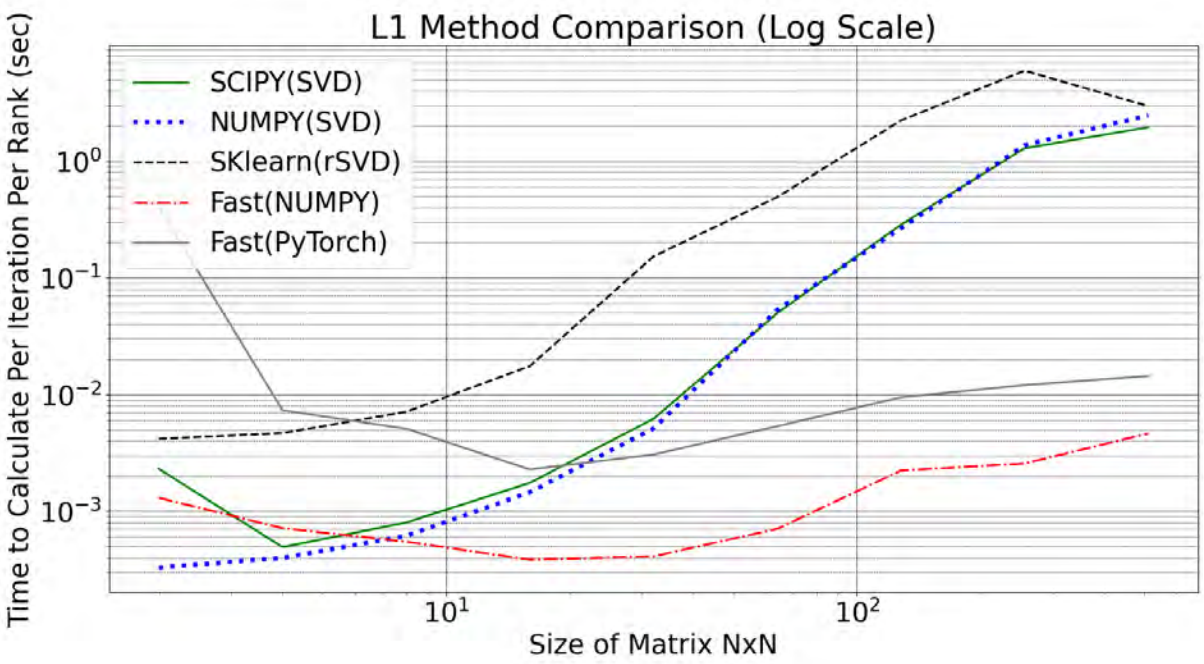


FIGURE 3.1. L_1 Results

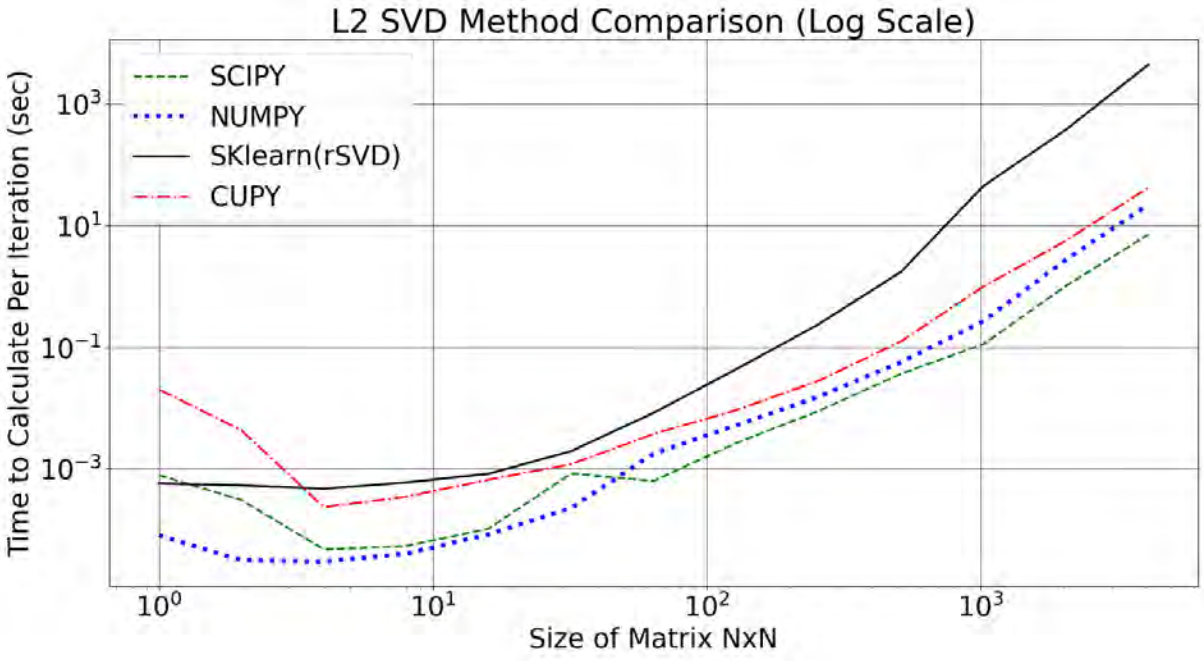


FIGURE 3.2. L_2 Results

TABLE 3.2. L_1 1035 Results Table

Size	L_1 Average Time to Calculate per Iteration per Rank (sec)				
NxN	SCIPY(SVD)	NUMPY(SVD)	SKlearn(rSVD)	Fast(NUMPY)	Fast(PyTorch)
2	0.0023059	0.0003294	0.0041629	0.0013015	0.4032737
4	0.0004921	0.0003980	0.0046698	0.0007160	0.0073261
8	0.0008000	0.0006147	0.0071136	0.0005480	0.0050910
16	0.0017491	0.0014551	0.0175244	0.0003840	0.0022782
32	0.0061976	0.0051233	0.1525485	0.0004100	0.0030585
64	0.0497812	0.0530894	0.4905794	0.0007064	0.0053540
128	0.2848567	0.2657645	2.2298846	0.0022272	0.0094400
256	1.2873271	1.3682969	5.9424476	0.0025635	0.0120314
512	1.9495564	2.4649807	2.9586152	0.0046342	0.0144317

TABLE 3.3. L_2 3284 Results

Size	L_2 Average Time to Calculate per Iteration (sec)			
NxN	SCIPY	NUMPY	SKlearn(rSVD)	CUPY
1	0.0007803	0.0000796	0.0005712	0.0200463
2	0.0003045	0.0000311	0.0005306	0.0042687
4	0.0000470	0.0000296	0.0004662	0.0002328
8	0.0000531	0.0000397	0.0005913	0.0003447
16	0.0001027	0.0000827	0.0008199	0.0006564
32	0.0008283	0.0002251	0.0019358	0.0011786
64	0.0006255	0.0017564	0.0082451	0.0037089
128	0.0026244	0.0051258	0.0428333	0.0091233
256	0.0088165	0.0151746	0.2355485	0.0276826
512	0.0362235	0.0563013	1.7141811	0.1232764
1024	0.1105798	0.2655464	44.3268075	0.9819591
2048	1.0058860	2.7159728	372.8572238	5.5728318
4096	7.0636545	21.7746839	4376.5408500	41.4346752

insight as to the advantages of the Fast method in most applications of L_1 PCA including dimensionality reduction. Comparing the gray line to the red line in 3.1 shows that the gap between CPU and GPU operation narrows as the matrix size increases. Running on the GPU means having to transfer the matrix to and from the GPU, adding time to the calculation.

Table 3.3 contains the results of over 3284 tests of the L_2 algorithms in parallel. Figure 3.2 is a plot of the numbers on a log scale to give better visualization of these results. The NUMPY and SCIPY based SVD algorithms calculate the PCs with little difference. The CUPY based method uses the GPU to calculate the SVD and again as the matrix size

increases the gap is closed between it and the of the other two high performing algorithms. The current implementation of the random SVD method in the SKlearns library underperforms in both the L_1 and L_2 implementations never falling in with the leaders even when the matrix grows to a very large size. The results of the L_1 method show rSVD moving closer to the results of NUMPY and SCIPY SVD methods. The further depth of investigation of the L_2 methods shows that the random SVD method curve arches up at a higher rate when compared to that of the other functions, this would lead to the inference that upon further testing and expansion to larger matrix size it would not completely close the gap. This is not conclusive and warrants further exploration. The inclusion of these results and this method was done as a additional experiment like the function itself.

3.4. Conclusion

The experiments of this study have shed light onto the different methods used to solve for L_1 and L_2 PCs using the current state of the art methods. These algorithms are commonly used to solve signal processing, machine learning, pattern recognition, and classification problems. The information is meant to guide the implementation of further studies into the use of the proper function in the solution to new problems. The solving of SVD based L_2 is reasoned by the results to be comparable for the SCIPY, NUMPY and CUPY method and implementation should be dependent across these libraries based upon other functions available and computational power contained in the hardware. The Fast method reliably performed significantly faster then that of the SVD based methods, this is due to the fact that the SVD method solves for all PCs every time while the Fast method iterates until it reaches the prescribed rank. The speed of the Fast method can be exploited in many applications of compression and identification. The use is the SVD method makes the most sense when a full rank result is requested, such as restoration and other procedures where computational costs are valued less in comparison to accuracy.

CHAPTER 4

ERRONEOUS DATA DETECTION: METEOROLOGICAL PCA

There are multiple applications for meteorological data. For instance, precipitation and temperature data can track the health of plants [8] or be used to create environmental models to predict disasters such as floods, droughts, and fires. The prediction of environmental changes through the use of signal processing techniques on data captured from multiple sensors carried out on models that have learned from previous data is the popular technique in use today. Common machine learning techniques for processing meteorological data employ decision trees, rule-based methods, neural networks, naive Bayes, Bayesian belief networks, and support vector machines. Accurate data is required for all of these methods to effectively create a precise model to simulate and predict at an acceptable level. Outliers and errors in the dataset can significantly skew the generated models that are relied upon by many sectors of society including agriculture, natural disasters, and meteorological forecasting. In this chapter, a method to identify and eliminate outliers from meteorological data to enhance the accuracy of models by applying a blind thresholding algorithm to the principal components (PCs) obtained from L_1 and L_2 norm Principal Component Analysis is presented. Data that has been stored is often subject to errors such as missing measurements and corrupted data points [53, 65]. Faulty hardware, signal degradation, and failing power sources of wireless sensor networks used to gather the information can introduce unreliable readings [81]. When introduced into the dataset, these errors will skew the accuracy of prediction models significantly. Options to identify and remove outliers include statistical and principal component analysis where the efficacy of each is highly dependent on the dataset's characteristics. Mean and standard deviation are the most often used statistical analysis methods to contour the dataset by removing large variances form the set [34, 59]. Application of PCA methods can characterize a sample of historic meteorological data from a geographic location with known accuracy [24]. These characteristics can then be used to infer the validity of previously unknown data. There exists a need to develop methods that

identify and reject outliers contained in datasets of meteorological data. A novel PCA based algorithm designed to reject outliers contained in datasets of meteorological data is presented in this chapter. The PCs are calculated for a single geographic location and then projected onto unknown data from the same location, resulting in a variance matrix. This matrix is then processed with a blind thresholding algorithm that removes outliers to increase the overall data integrity.

4.1. Outlier Detection and Removal

The introduction of transient noise into a meteorological dataset creates erroneous data in an assumed error free set. Noise is injected at varying levels simulating corruption of the reading of each distinct sensor for a set number of days. To more accurately represent the degradation that can be introduced during the sensing, transmission and storage of information, a range of values are chosen.

Application of PCA techniques preserves the differences and removes the similarities from a set of correlated data. Simulations were pursued using both L_1 and L_2 PCs. The SVD method and single bit flipping algorithms of Chapter 3 are used to find the principal components.

The PCs represent the most important characteristics of the dataset. When projected onto unknown data, PCs highlight discrepancies between the two. Additional processing can be executed to obtain reliability weights [13] that are fed into a blind thresholding algorithm that selectively identifies and discards outliers from the original dataset based upon the normalized reliability weights (w_n^p).

$$w_n^p = \frac{r_n^p}{\Sigma_n r_n^p} \tag{20}$$

where

$$r_n^p = \|\mathbf{X}_n^p - \mathbf{Q}^p \mathbf{Q}^{pT} \mathbf{X}_n^p\|_2^{-2} \tag{21}$$

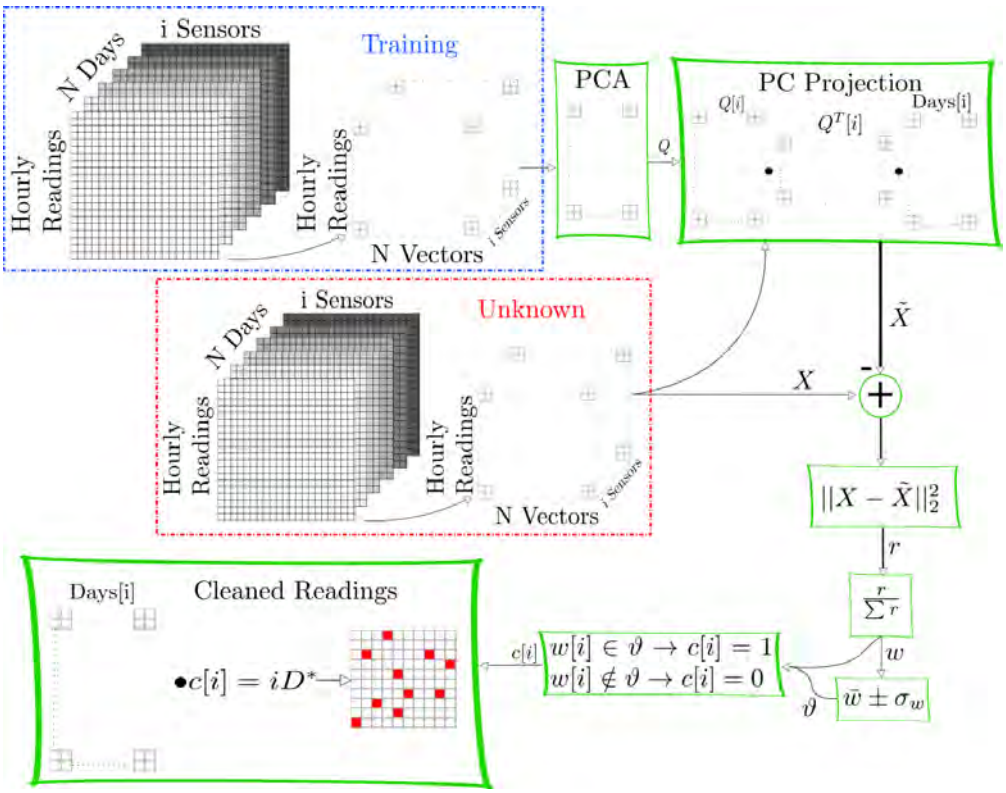


FIGURE 4.1. Novel Outlier Removal Algorithm

4.2. Implementation

Pictured in 4.1 is the implemented algorithm. Preprocessing of the data to sort and organize the sensor readings into vectors of daily readings is carried out first. The mean of each vector is then removed to obtain a metric of variance between individual sensor values. Subsets of the original dataset are used to create training sets for the algorithm. These sets are then characterized through PCA, by processing the data through the corresponding L_1 and L_2 algorithms. The obtained principal components (Q) are then passed into a novel outlier rejection algorithm used to identify the validity of the information contained in the unknown dataset. During the operation of the rejection algorithm, PCs (Q) are projected onto the unknown dataset, producing a matrix of similarity. A weighted distribution reliability index is calculated for each day at each sensor in the test set. Data validity analysis is then carried out utilizing a blind thresholding function, that uses the statistical variance of the unknown set with no inference from the training set.

4.3. Validation

Testing the performance of the algorithm, with known data that is corrupted to simulate erroneous information, is carried out over multiple experiments. The original dataset is randomly divided into training and testing sets that will serve as the unknown data for each experiment. Quantifying the results of the validation testing, the analysis of the resulting truth matrix corresponding to sensor readings is XORed with that of the mask used to create the unknown corruption set. To calculate the overall percent identification error, the number of false identifications returned are counted and then divided by the total number of samples.

$$PercentValidity = 100 \cdot (1 - \frac{F\_P + F\_N}{Total\_num\_Samples}) \tag{22}$$

4.4. Results

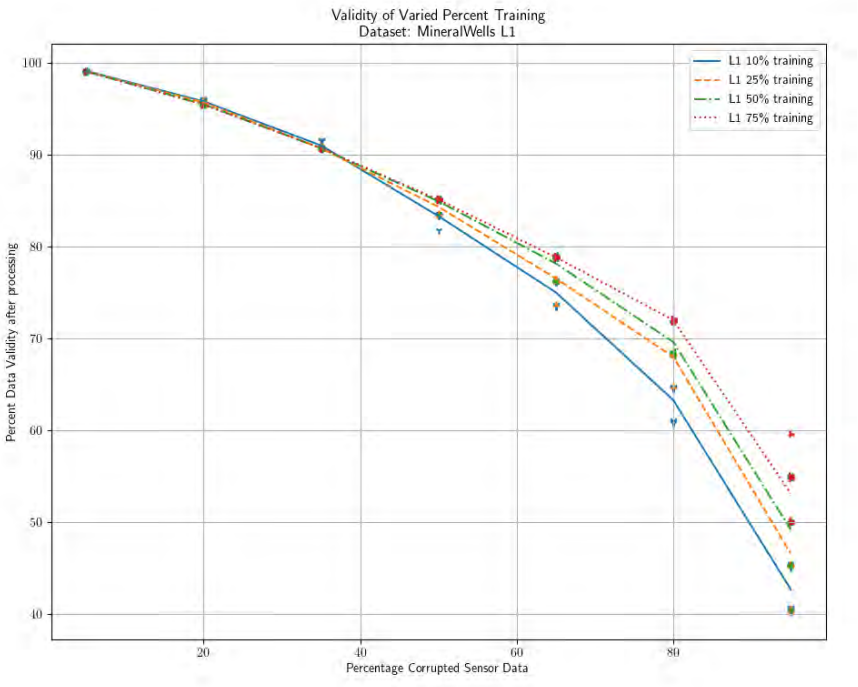


FIGURE 4.2. L1 Results of Corrupted Data Mineral Wells, Texas

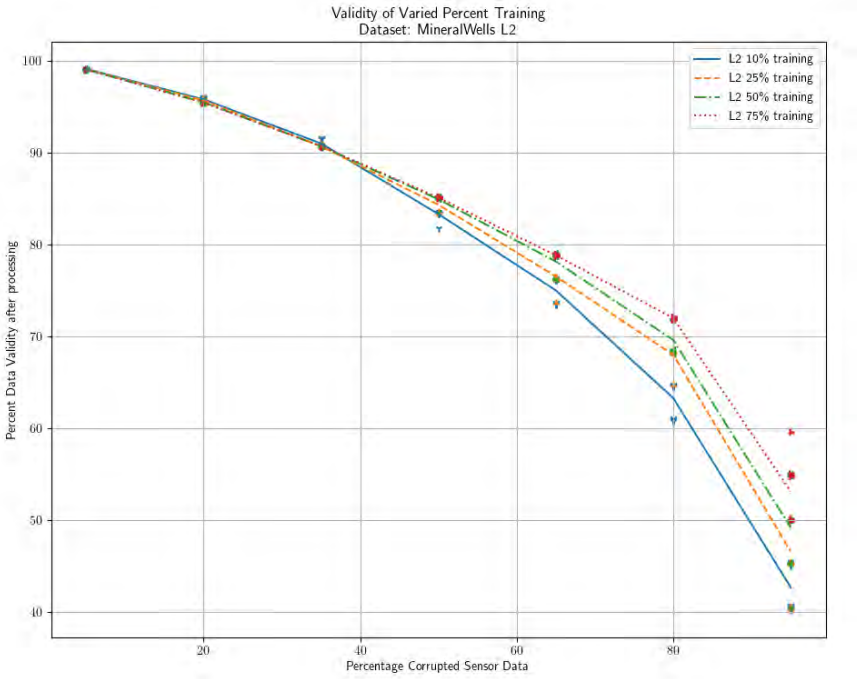


FIGURE 4.3. L2 Results of Corrupted Data Mineral Wells, Texas

Validation Datasets Historical Contents		
Station	Days with 24 Readings per Day	Year First Active
Mineral Wells, TX	13,525	1953
Meacham - Fort Worth, TX	8,100	1946
Love Field - Dallas, TX	15,418	1949
Dallas, TX	17,048	1941

TABLE 4.1. Validation Historical Dataset

Analysis of data from the national climate data center (NCDC) climate data online (CDO) database was carried out on sets of data from specifically selected locations that had reliably logged data at 1 hour intervals across 6 sensors measuring: Hourly Dew Point Temperature, Hourly Dry Bulb Temperature, Hourly Relative Humidity, Hourly Sea Level Pressure, Hourly Station Pressure, Hourly Wet Bulb Temperature, and Hourly Wind Speed. Table 4.1 contains the number of days for each dataset with 24 hourly record values and the year of first recorded reading for each station.

Figures 4.2 and 4.3 are the result of data processed from the Mineral Wells, Texas

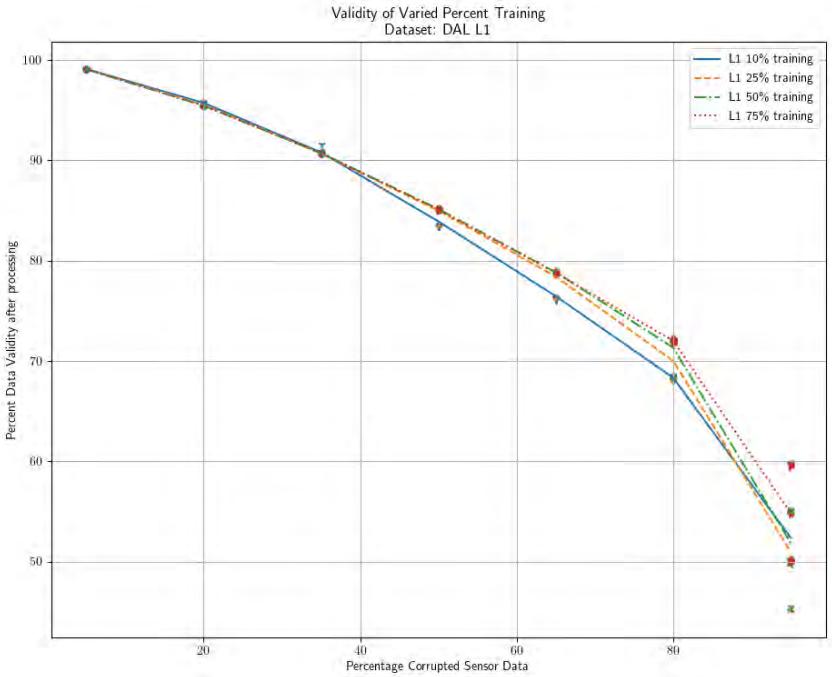


FIGURE 4.4. L1 Results of Corrupted Data Love Field Dallas, Texas

station. They present the outlier rejection at varied training set sizes as a proportion of the total data. Note the similarity of the L_1 and L_2 results in the plots demonstrating the closeness in accuracy of both PCA approaches as applied through the blind thresholding algorithm. The proposed solution is able to identify and remove outliers, even for large amounts of corruption. Figures 4.4 and 4.5 provide further confirmation of the results by demonstrating the application of the algorithm for Dallas, Love Field. The experimental results of Fort Worth, Meacham airport are plotted for L_1 in Figure 4.6 showing the most aggressive training roll off for lower rates as compared to higher training sizes, the results are almost identically mirrored in Figure 4.7 for the results of L_2 PCA. The final dataset under experimentation was that of Dallas, DAL with results similar to Love Field and Mineral Wells, experiencing less of a spread as the percentage corruption increases, displayed in Figures 4.8 and 4.9 for L_1 and L_2 respectively.

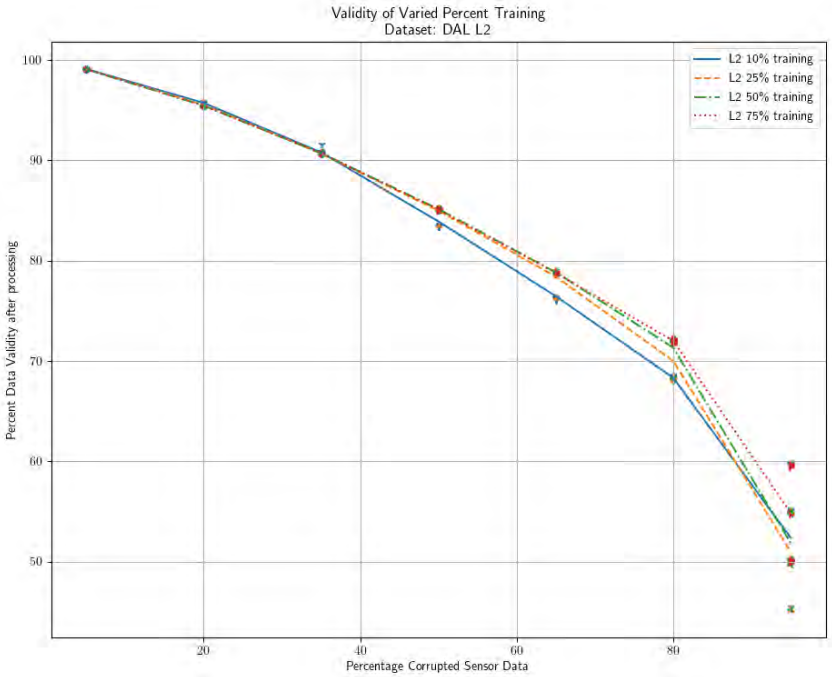


FIGURE 4.5. L2 Results of Corrupted Data Love Field Dallas, Texas

4.5. Conclusion

Accurate historical meteorological data is paramount in the prediction of environmental models. The predictions influence many industries including agriculture and energy, as a key variable in their projection analysis of production and demand. The application of the outlier rejection algorithm enables unknown meteorological data gathered at previously characterized locations to be filtered into a more accurate dataset. At only 10% of the data or as little as 800 days of readings for training, the algorithms hits over 50% validity at 90% corruption and 95% validity with 20% corruption. With the added clarity of information, predictive models can better infer future conditions leading to a longer time basis of accuracy. Application possibilities are numerous from traditional logged data sources operated by governments to crowd sourced sensor data collected on mobile devices. The need for accurate information is ever growing as technology advances the drive for innovation in every industry.

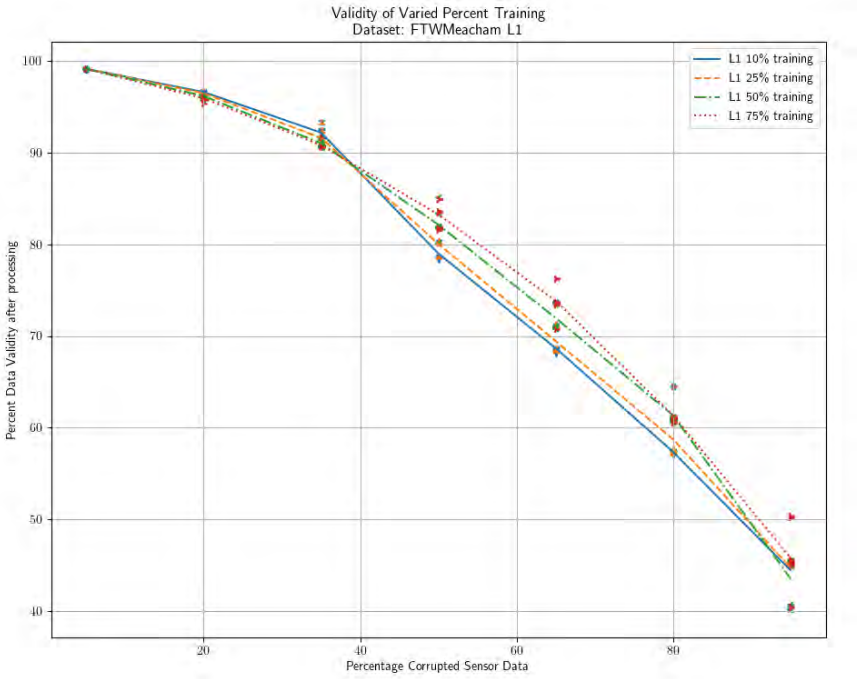


FIGURE 4.6. L1 Results of Corrupted Data Meacham Fort Worth, Texas

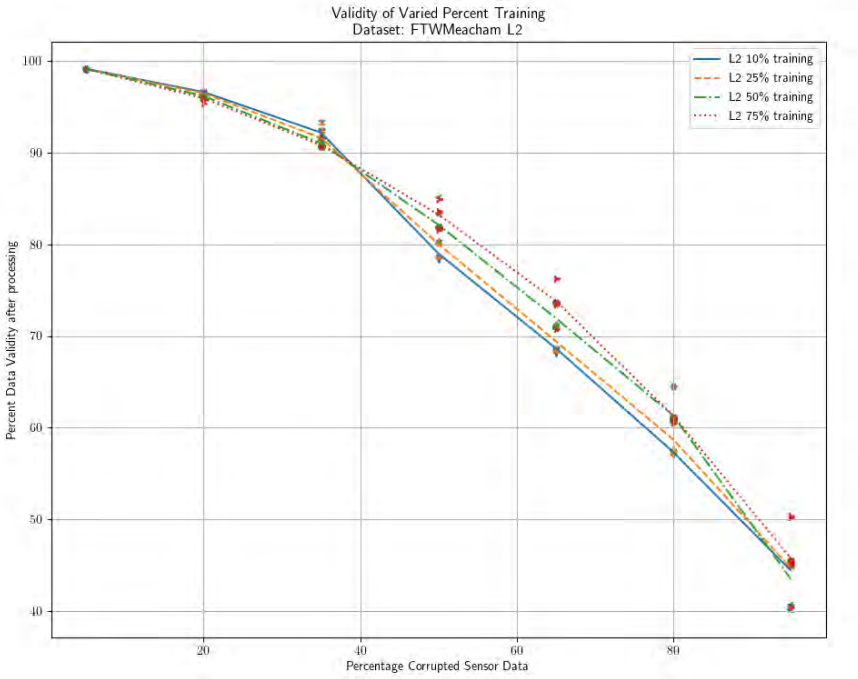


FIGURE 4.7. L2 Results of Corrupted Data Meacham Fort Worth, Texas

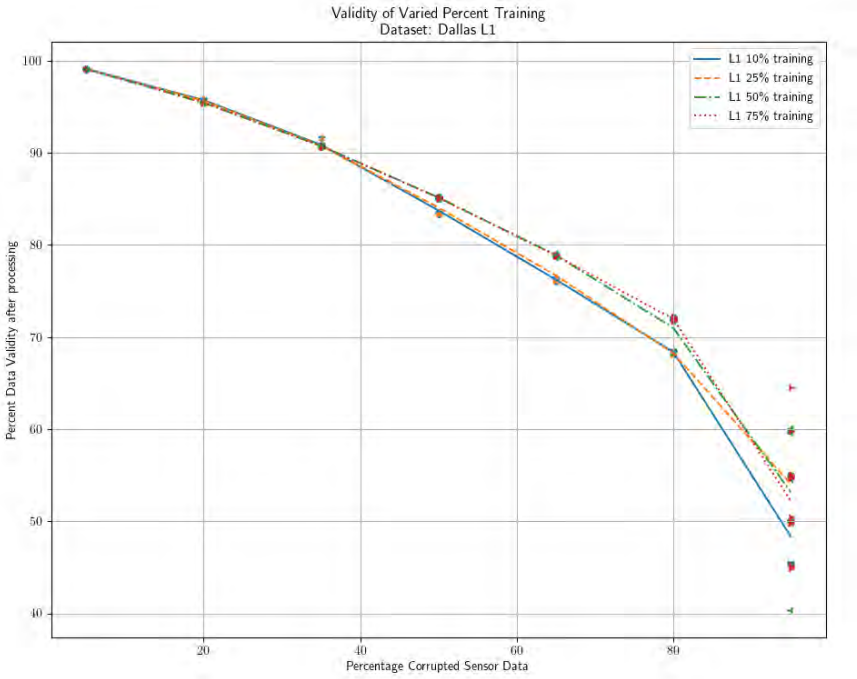


FIGURE 4.8. L1 Results of Corrupted Data Dallas, Texas

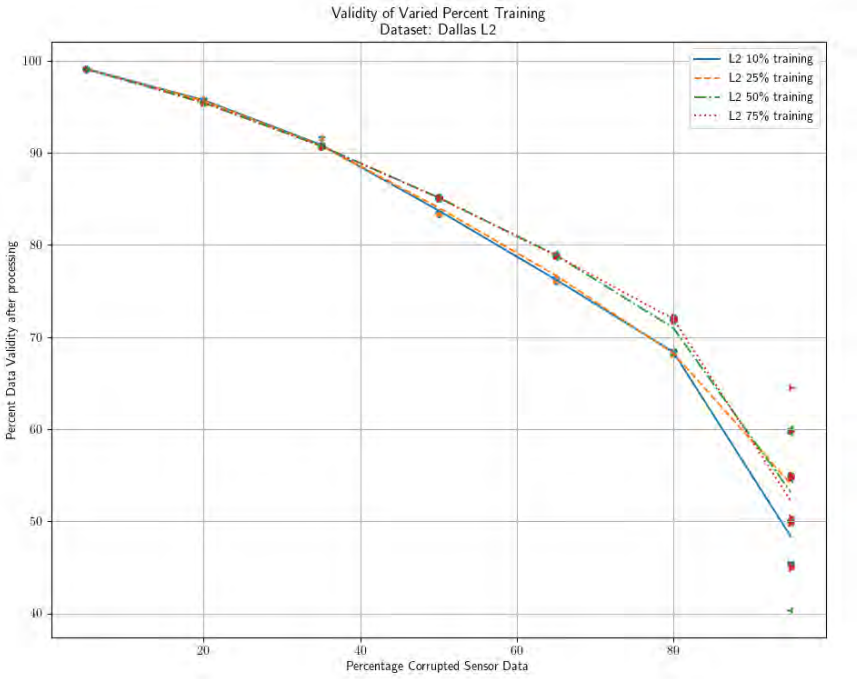


FIGURE 4.9. L2 Results of Corrupted Data Dallas, Texas

CHAPTER 5

MULTI-BAND IMAGE FUSION IN THE PRESENCE OF NOISE

Multispectral satellite image fusion is important to provide an enhanced visual representation. In the past, there have been many different methods utilized for feature extraction and fusion of terrestrial satellite images. These techniques range from methods of classical signal processing [41] to applications of neural networks [76]. Each method aims to achieve a similar goal but with many different associated costs concerning the balance between performance and detail. The use of a common signal processing techniques of up-sampling and bi-linear interpolation are frequently used to fuse images together but can often blur areas of detail [36]. Image fusion techniques typically rely on IHS transforms, PCA, or Gram Schmidt method. With its superior resistance to color distortion and lower spectral distortion compared to IHS [62, 83], PCA also removes the redundancy from a set of correlated data while preserving the variance, making it a superior choice for image fusion. There have been other accounts of machine learning methods used to fuse multispectral data including residual neural networks and convolutional neural networks [3]. Residual neural networks simplify the layers by skipping or making jumps within, these skip functions offer a boost in learning performance, without the issue of oversaturation. Image detail is lost when applying this technique to high resolution multispectral images as the function jumps between weighted layers [61]. Deep convolutional neural networks have also been used to achieve a very fine level of detail during image fusion. The associated computational cost results in a high increase in processing time due to the complexity of the dataset [37]. Denoising neural networks have been developed using autoencoder architecture and have become popular tools to correct images [64, 44]. The recent advances have created robust models for a range of noises and allowed for blind implementation [1]. The proposed method seeks to optimize the balance between neural networks and PCA, drawing upon the strengths of each, utilizing PCA to identify the outliers and pass the samples in question to a neural network for denoising before final fusion. Developing further on the works completed by King et al. [32],

we propose a L2 PCA method that replaces their controlled fusion algorithm with that of a neural network. This paper seeks to deliver an engine that removes significant out of range values while emphasising information that is deemed important. The method recognizes corruption contained in the input data and outputs a cleaned version of an image to be used in PCA based image fusion.

5.1. Corruption

Salt and pepper noise is introduced into the satellite sensor data to simulate damaged pixel elements in the satellite's sensor [7]. Salt and peppering appears in the image as the floor (pepper) and ceiling (salt), values of 0 and 255 respectively for the 8-bit color space image used in this paper. A set number of images within the dataset are chosen at random to be corrupted with varying levels of percentage pixels corruption and salt to pepper ratio.

5.2. Image Fusion

Image fusion seeks to create a combination visualization of sensory data to effectively communicate the information contained within, heightening the important data and muting the erroneous. From the literature, principal component based fusion has shown superior results as opposed to other methods [67, 71, 50]. Exploration of image fusion is performed using both L_1 and L_2 norm PCA. The methods presented are developed based upon the implemented L_1 and L_2 PCA methods of Chapter 3, specifically those of single bit flipping for L_1 and indexed SVD for L_2.

5.2.1. Image Fusion Algorithm Without Control

The method presented by Chamadia and Pados is used as a benchmark for comparing the proposed algorithm performance [13]. In that work, a weighted PC reliability summation algorithm is implemented to accomplish fusion of the data set. The calculated PCs ($\mathbf{Q}^p$) of the vectorized images ($\mathbf{X}_n^p$) are used to calculate the reliability weight r_n^p.

$$r_n^p = \|\mathbf{X}_n^p - \mathbf{Q}^p\mathbf{Q}^{pT}\mathbf{X}_n^p\|_2^{-2}, \quad n = 1, 2, \cdots, 14 \tag{23}$$

The reliability weight is then further processed to obtain the normalized reliability weight (w_n^p).

$$w_n^p = \frac{r_n^p}{\Sigma_n r_n^p} \tag{24}$$

These weights are then applied as weighted sums to the vectorized image matrix that is then used to form the fused image.

5.2.2. Controlled Weighted Fusion

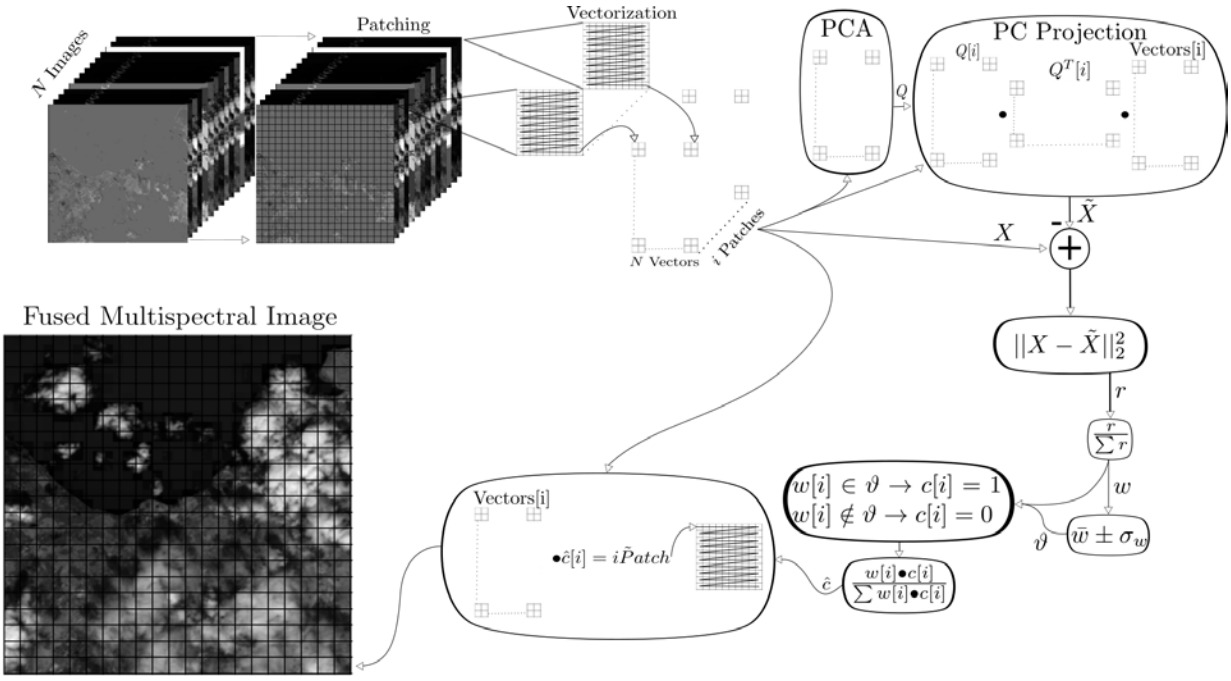


FIGURE 5.1. Controlled Weighted Principal Component Analysis Fusion Algorithm Overview

The controlled image fusion process shown in Figure 5.1 begins with a set of N multispectral images $A_n\,[i_1, \ldots, i_n]$. Various amounts of corruption are introduced into the set of images through salt and pepper noise. The noise is added to randomly selected images where the amount of noise introduced is specified by the percent of pixels corrupted and the salt to pepper ratio. The non-corrupted and corrupted datasets are then divided into smaller non-overlapping sub-sections or patches. These patches are flattened into vectors, where the i^{th} vector of the n^{th} image is stored into a matrix $DatasetPatched$. This new

matrix contains the vectors of similar data contained in the blocks of each image that will be used as the training set where each vector represents the different spectral data taken from the same block index location [26].

Algorithm 9 Dynamic Thresholding Algorithm

1: **Inputs:**
   $||w||$
2: $Thresh \leftarrow Mean(||w||) - Std(||w||) * Tol$
3: **if** $||w|| < Thresh$ **then**
4: $||w|| = 0$
5: **else**
6: $||w|| = 1$
7: **end if**

The resulting matrix is then processed by L_1 and L_2 algorithms to obtain the principal components. The resulting PCs are stored in a new matrix that is passed to the dynamic thresholding control algorithm detailed in Algorithm 9. This controlled fusion is achieved by not fusing the image at the weighted levels but through a dynamic thresholding algorithm. The threshold is set by subtracting the mean and then multiplying the tolerance setting by the standard deviation of $||w||$. The $||w||$ and the weight are adjusted to either include or exclude the sample channel from the image. The remaining channels are then compiled into the the final fused image.

5.3. Data

The data used to test the proposed fusion algorithm is sourced from The European Space Agency (ESA) Sentinel Online database [18], which provides satellite images ranging across 12 spectral bands, as shown in Table 5.1. This provides multiple representations of a given area for each range on the spectrum, potentially revealing details exclusive to a single band.

5.4. SSIM Metric

There is no standardized reference dataset to evaluate the efficacy of fusion. Quality metrics can be used to establish the relationships between the original and fused results, but are often found to be a suboptimal index of meaningful visual attributes [62]. To best

Band	Resolution	Central Wavelength	Description
B1	60 m	443 nm	Ultra blue (Coastal and Aerosol)
B2	10 m	490 nm	Blue
B3	10 m	560 nm	Green
B4	10 m	665 nm	Red
B5	20 m	705 nm	Visible and Near Infrared (VNIR)
B6	20 m	740 nm	Visible and Near Infrared (VNIR)
B7	20 m	783 nm	Visible and Near Infrared (VNIR)
B8	10 m	842 nm	Visible and Near Infrared (VNIR)
B8a	20 m	865 nm	Visible and Near Infrared (VNIR)
B9	60 m	940 nm	Short Wave Infrared (SWIR)
B10	60 m	1375 nm	Short Wave Infrared (SWIR)
B11	20 m	1610 nm	Short Wave Infrared (SWIR)
B12	20 m	2190 nm	Short Wave Infrared (SWIR)

TABLE 5.1. Spectral Bands of Sentinel 2A and 2B

evaluate the outputs of the two algorithms, the uncorrupted fused images are evaluated against the corrupted fused outputs through the structural similarity (SSIM) index measure. SSIM is a numerical method for determining the similarity between two images. It is based on the combination of the luminance (l), contrast (c), and structure (s) comparison, as defined by the local mean, variance, and covariance of the images ($\mu_x, \mu_y, \sigma_x, \sigma_y, \sigma_{xy}$) [4].

$$\text{SSIM}(x,y) = \left[l(x,y)^{\alpha} \cdot c(x,y)^{\beta} \cdot s(x,y)^{\gamma}\right] \tag{25}$$

$$\begin{aligned} l(x,y) =& \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \\ c(x_y) =& \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} \\ s(x,y) =& \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3} \end{aligned} \tag{26}$$

5.5. PCA Fusion Testing

Validation studies were executed to asses the performance of the proposed controlled weighted fusion algorithm on multispectral satellite images taken from Sentinal 2A at 60m resolution as shown in Figure 5.2. For comparison the weighted fusion algorithm was tested in parallel giving a basis for analysis.

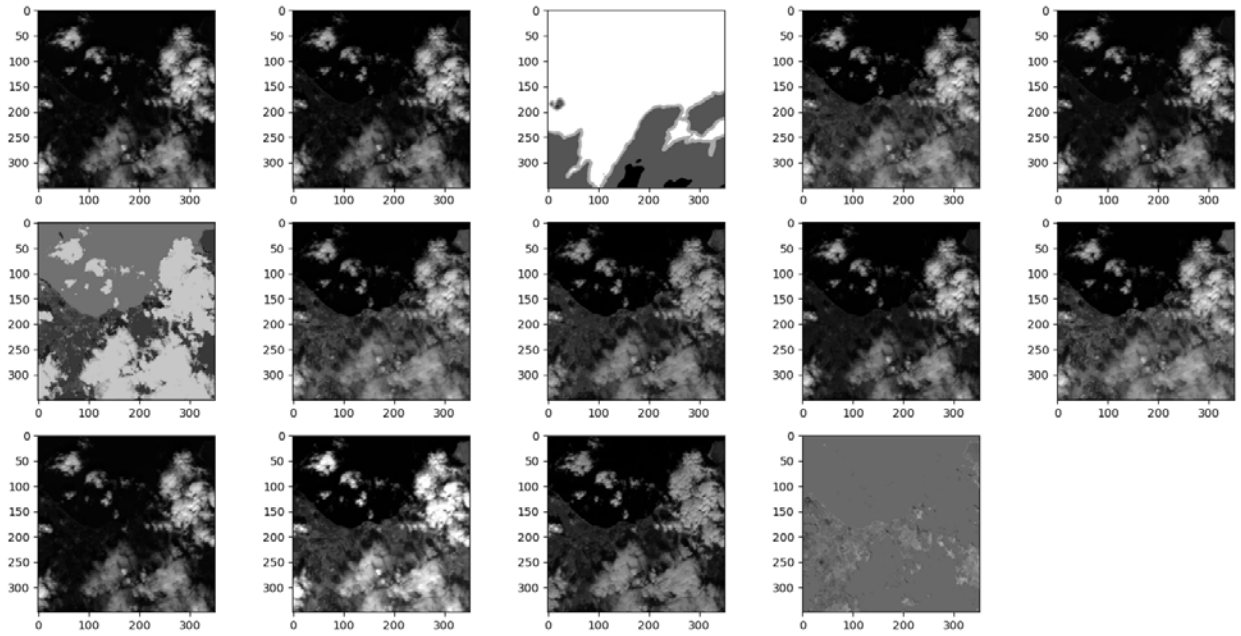


FIGURE 5.2. Sentinel 2A 60m Resolution Images (from top left to bottom right): B01, B04, AOT, B06, B03, SCL, B07, B11, B05, B8A, B02, B09, B12, WVP

Direct fusion of the clean (uncorrupted) images for both L_1 and L_2, as seen in Figure 5.3, are respectively used in calculating the SSIM performance of the proposed method compared to the existing weighted fusion method. The top and bottom fused images on the left are the result of using the method from Section 5.2.1. The top and bottom right images are processed using the proposed algorithm with dynamic thresholding. To the human eye, the images fused using L_2 (top) look the same as the corresponding L_1 images (bottom) for both the proposed algorithm and the previous method. The controlled method presented in this section produces a visibly clearer image as seen on the right in Figure 5.3.

Figure 5.4 displays an instance of the resulting corrupted input data when two images are randomly chosen to add corruption. In this case, bands B04 and SCL have 95% of their pixels are corrupted by 50% ratio salt and pepper noise. The fusion of the corrupted images is shown in Figure 5.5. Similar to the uncorrupted case, the proposed method produces a visually superior result and there is no noticable difference between L_1 and L_2.

The SSIM values are plotted versus varied corruption parameter in Figure 5.6. The data was generated by changing the number of images that were corrupted. Each randomly

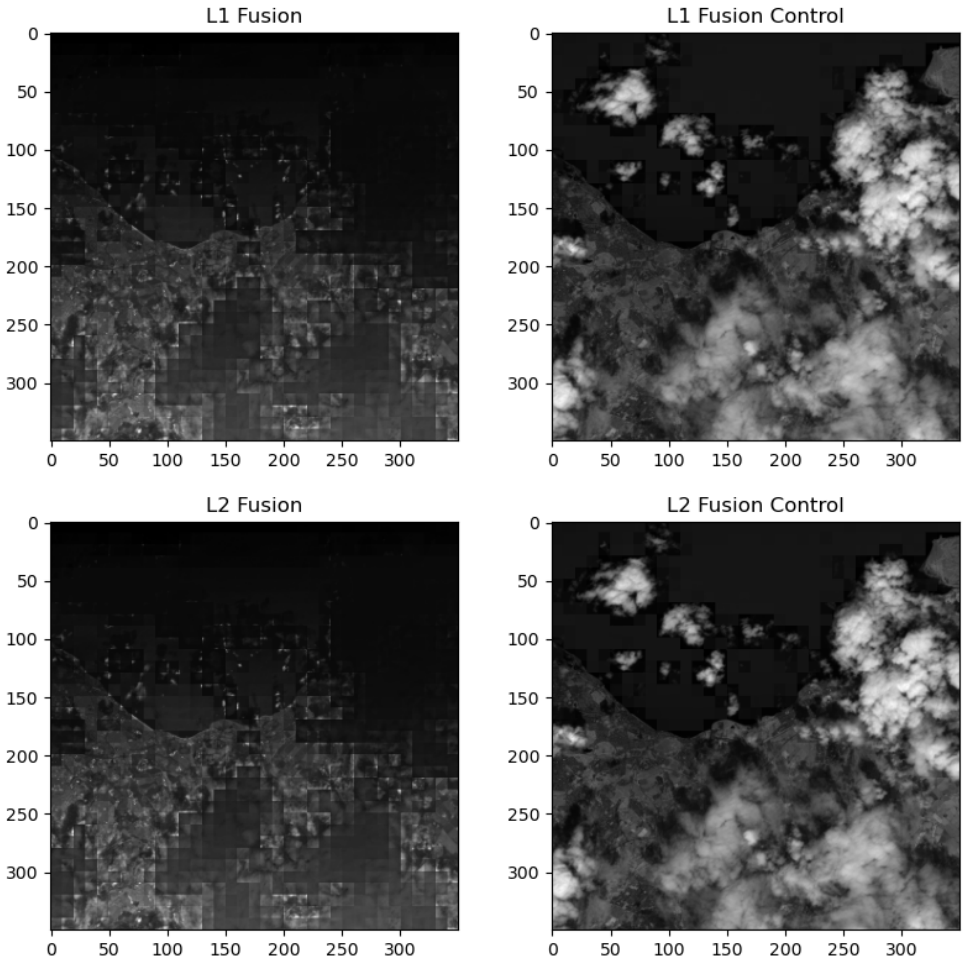


FIGURE 5.3. L_1 and L_2 Fusion of Uncorrupted Sentinel 2A Images

corrupted image suffered from 50% salt and pepper noise applied to 95% of pixels. The SSIM calculated when using the L_1 or L_2 weighted fusion algorithm deteriorated more quickly than the proposed controlled algorithm as the number of corrupted images increased. Even with high levels of corruption (over 50% of images corrupted), the method presented in this thesis is able to provide a fused image that is at least 70% similar with the uncorrupted fused image.

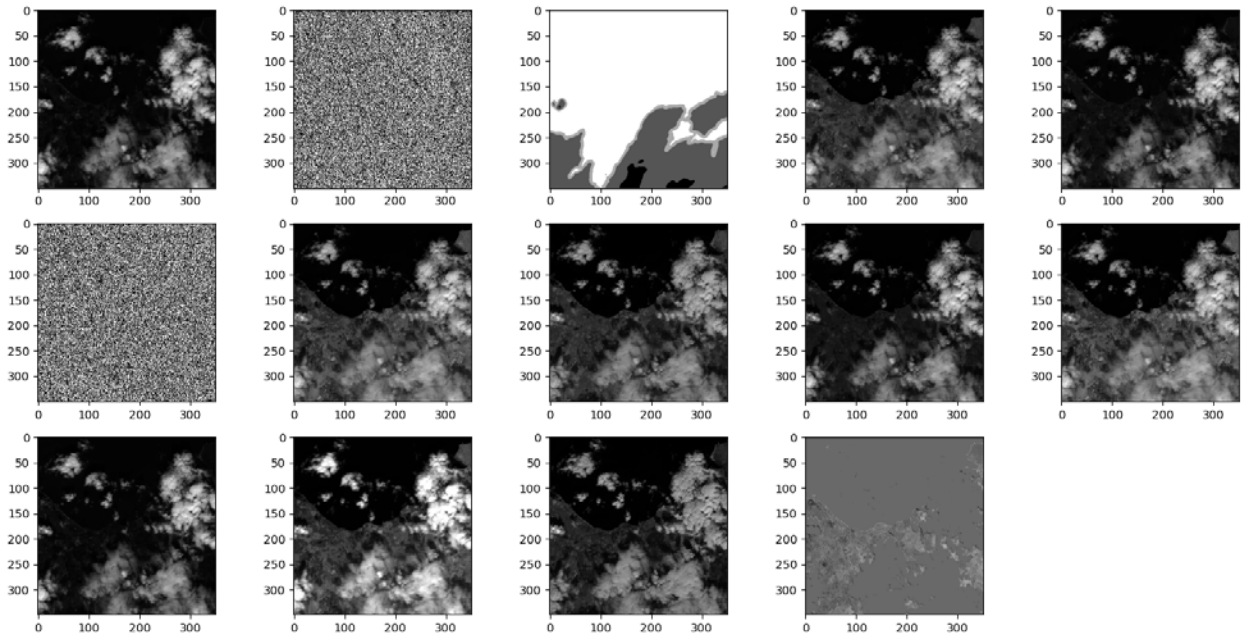


FIGURE 5.4. Example Corrupted Sentinel 2A Images: 2/14 Images 95% Corrupted by 50% Salt and Pepper Noise

5.6. Anaylsis of L_1 and L_2 Controlled Weighted Fusion

The controlled weighted fusion algorithm presented is able to successfully combine multi-band satellite images, improving the visualization for a wide range of corruption. The PCA based method is implemented using both L_1 and L_2 norms. Simulations verify the efficacy of the procedure showing similar results of both of the PCAs integrated into both of the fusion algorithms in question. The controlled weighted fusion algorithm is able to reduce the impact of corrupted data on the fused image resultant.

5.7. PCA with Autoencoder Implementation

As documented in 5.6, the presented results from both L_1 and L_2 PCA techniques to fuse multispectral images are similar. Therefore, we will utilize an L_2 based controlled fusion algorithm for comparison in the remainder of this chapter. As found in Chapter 3, the L_2 SVD based method is computationally faster on the devices at hand when computing the full rank solution as opposed to the tested L_1 methods. With the results from the previous section now recorded, modifications are carried out on the corruption of the image using a custom corruption algorithm.

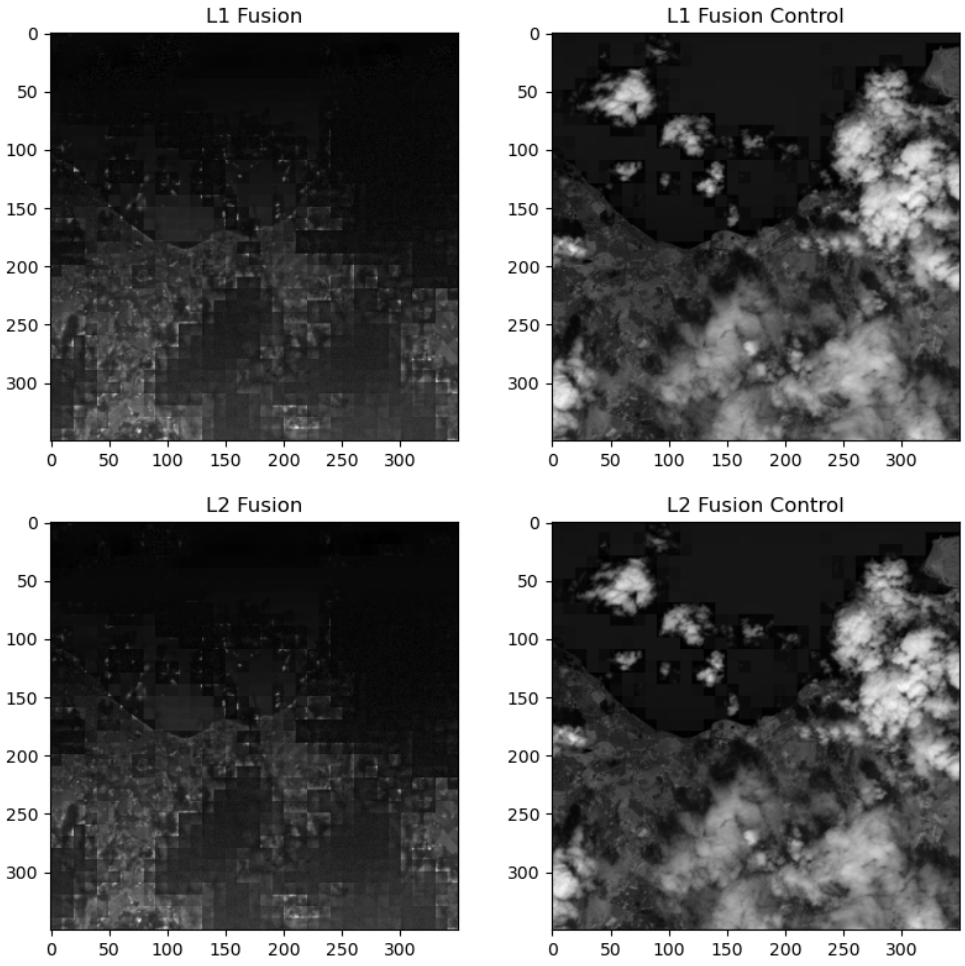


FIGURE 5.5. L_1 and L_2 Fusion of Corrupted Sentinel 2A Images.

5.7.1. A More Advanced Corruption

In order to test the performance of the proposed autoencoder fusion network, synthetically corrupted datasets are necessary. To adequately represent real world image corruption in a synthetic form, a new method is developed to account for the dense nature of the multispectral data provided by ESA. When using a neural network to detect out of range values while correctly weighting the important components, special consideration of gradient descent methods utilized is necessary. Introducing a multispectral image with a band entirely

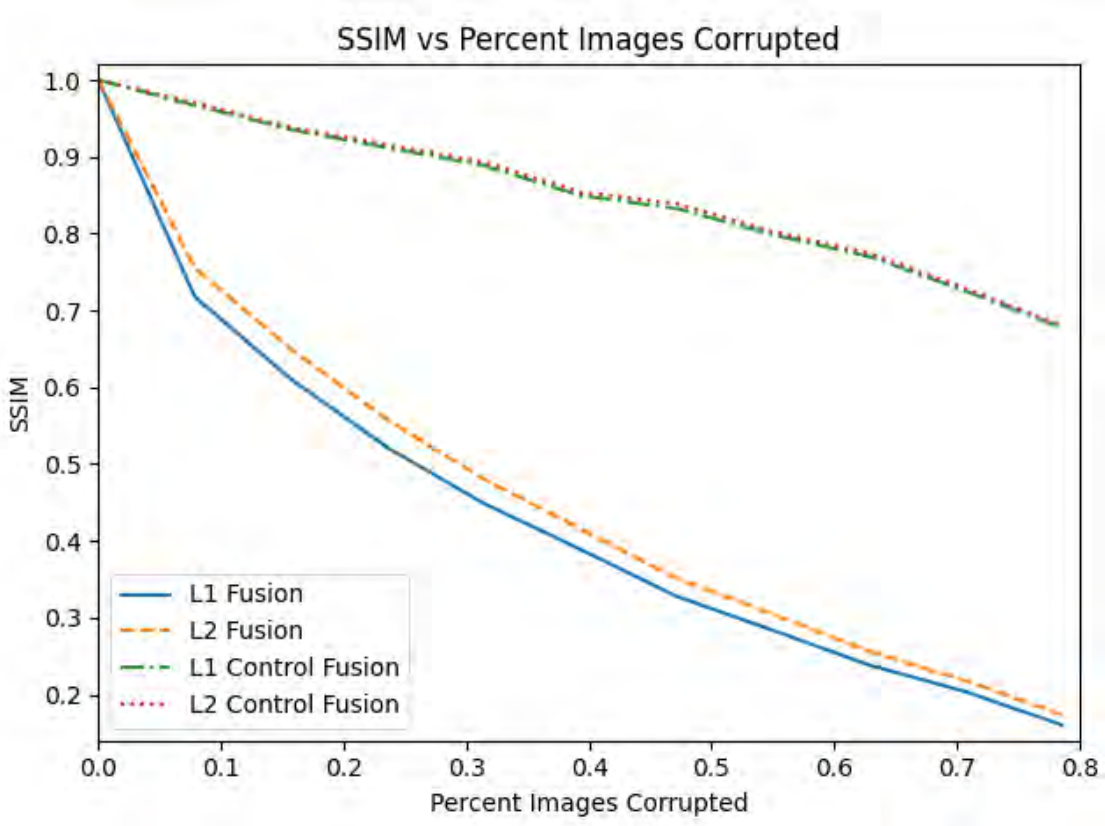
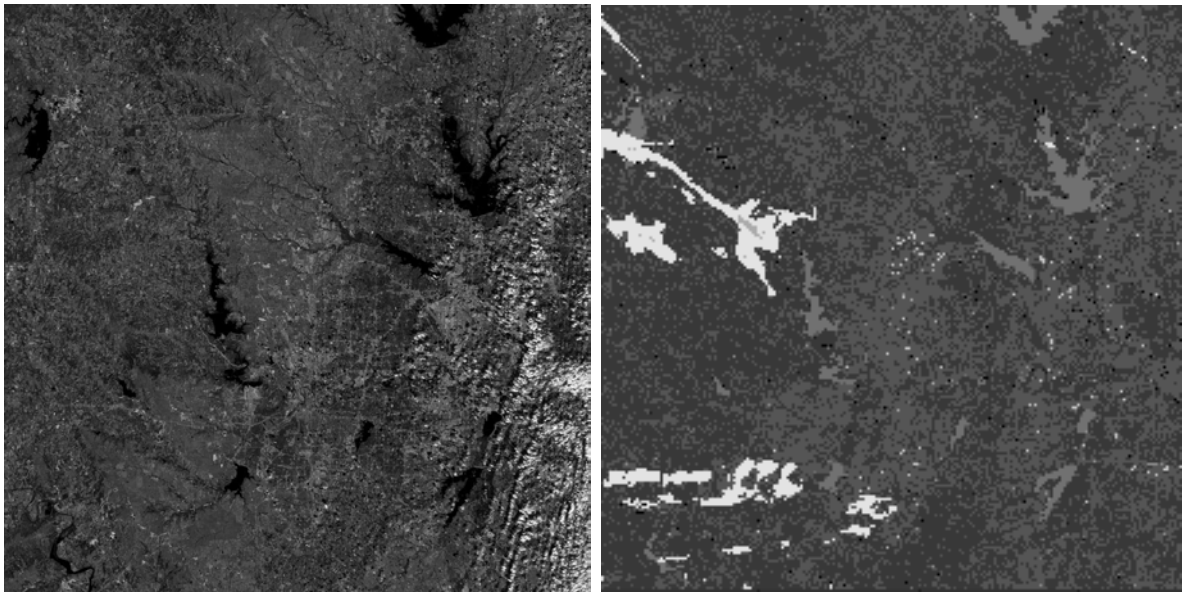


FIGURE 5.6. SSIM Values for Different Numbers of Corrupted Images

synthetically corrupted would haphazardly contaminate the results of the loss function, potentially removing important information from the output when future datasets are used. This risk is reduced by imitating corruption methods induced from real world examples.

Often times corruption in an image is found in groups rather than an equal distribution. Along with the methods of PCA based image fusion, an algorithm to synthetically corrupt images for testing to a user-specified percentage that would mimic any real world incidental corruption was created. An example of the applied synthetic corruption can be seen in Figure 5.7. This form of geometric corruption imitates a random distribution of noise throughout each slice, without losing complete integrity of the original image. Training of the denoising network on the noisy images allows the network to experience a variety of different possible inputs, all systematically worse than what would be encountered by typical sensors collecting multispectral data. This provides the ability to measure the corruption



(A) Fused Image (B) Corruption after Fusion

FIGURE 5.7. Sample Fusion of Images of Sentinel 2A and 2B

removal efficacy.

5.7.2. Autoencoder System Model

The composite system model is illustrated in Figure 5.8, implemented in a modular way for testing and tuning. The large amounts of data contained in multispectral images are intelligently combined to create a clear and vivid visualization of the information. Through the implementation of an autoencoder, the noise is removed from the data to increase the performance of the image fusion algorithm. The neural network architecture in Figure 5.9 is designed to be efficient and relies on convolutional layers which have been shown in previous works to perform extremely well when applied to image denoising.

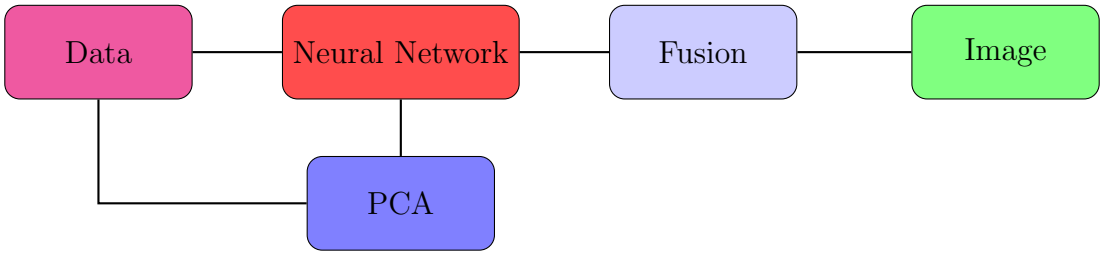


FIGURE 5.8. Learning System Design

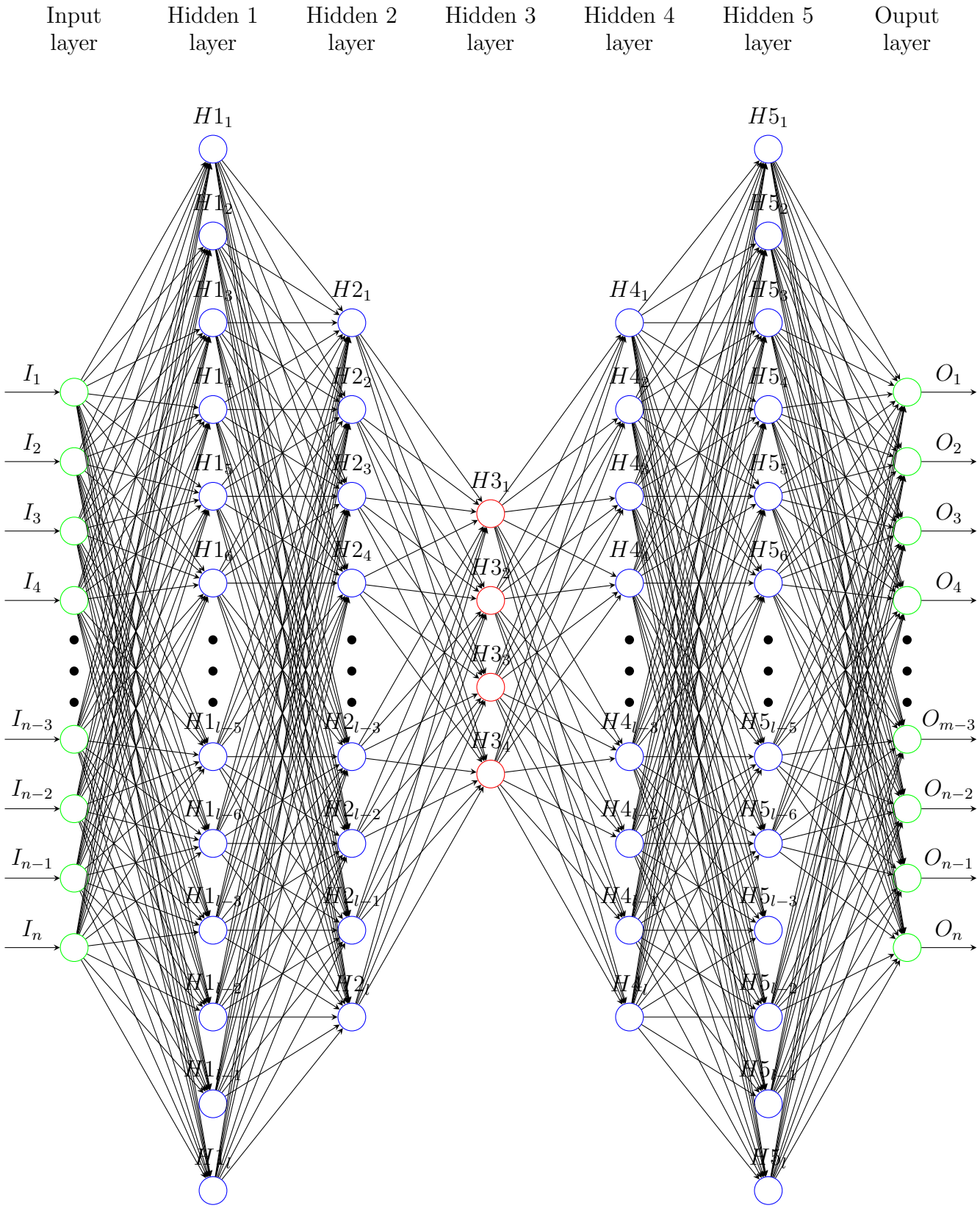


FIGURE 5.9. Neural Network Architecture: Autoencoder

Algorithm 10 2D Convolution Auto Encoder

Inputs:
    2D Image
Convolution 2D 64 x 3 x 3
Convolution 2D 64 x 3 x 3
Convolution 2D 32 x 3 x 3
    Max-Pooling 2D 2 x 2
Convolution 2D 32 x 3 x 3
Convolution 2D 32 x 3 x 3
    Max-Pooling 2D 2 x 2
    Encode
Convolution 2D 32 x 3 x 3
Convolution 2D 32 x 3 x 3
    Upsample 2D 2 x 2
Convolution 2D 32 x 3 x 3
Convolution 2D 32 x 3 x 3
Convolution 2D 64 x 3 x 3
Convolution 2D 64 x 3 x 3
    Upsample 2D 2 x 2
Convolution 2D 3 x 3
    Decoded
Output:
    Denoised 2D Image

5.7.3. Weighted Fusion Autoencoder

The weighted fusion algorithm presented in 5.2.2 is implemented up until the point of fusion, at which time the weights and original images along with the calculated PCs are analyzed by a control algorithm and selectively corrected through an autoencoder before being fused into the final image.

The autoencoder is constructed with CNN layers, comprised of 2-D convolutional layers, 2-D max pooling layers, and 2-D up-sampling layers implemented in combination. The autoencoder neural network can be viewed as two networks that feed into each other. The first network takes the image and reduces its dimensionality as it progresses through subsequent hidden layers. The second network takes the resulting information from the reduced dimensions and creates and output that matches the size of the original input image, preserving image resolution while performing non-linear processing. Figure 5.10 shows a 3D visual representation of the autoencoder neural network architecture.

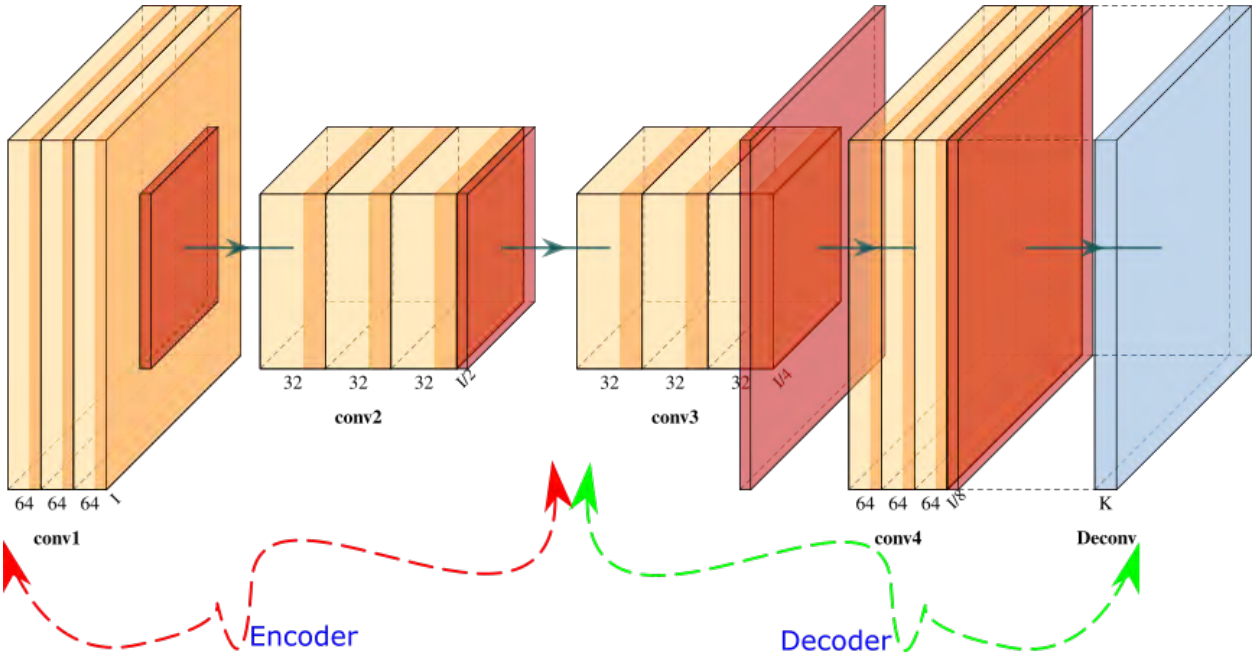


FIGURE 5.10. 3D Rendering Layered Block Representation of the 2D Autoencoder Neural Network

The optimizer employed in the autoencoder network is the 'ADAM' algorithm [33], combining the benefits from the adaptive gradient descent algorithm and root mean square propagation. The main advantage that the 'ADAM' optimizer offers our network is that the algorithm varies the learning rate. This in turn varies how much each node in the network will change its weights for their activation function, rather than keeping the learning rates fixed as is done in the typical gradient descent algorithm used in many neural networks.

5.8. Autoencoder Results

SSIM was used as an accuracy metric to measure performance of the autoencoder network. Utilizing this metric, the SSIM between an image from the corruption algorithm and the output of the fusion algorithm can be compared against the clean input image. This process not only provides a quantifiable measure of the effect of the corruption to ensure it reaches the specified corruption level, but also assesses how well the neural network cleans the image prior to completing the fusion. Testing the corruption algorithm and autoencoder neural network at various corruption levels is plotted in Figure 5.11. The blue line represents the SSIM comparison of the corrupted image to the original and the orange line shows

the comparison of the cleaned output image to the original, confirming the design validity. Displayed in the graph, the average noise induced by the corruption algorithm is around 20%, with the cleaned output image showing a reduction in noise down to an average of 2-3% per image, with a max noise level of approximately 5% correlating with the maximum noise levels fed into the algorithm.

5.9. Results

Testing of the entire system was carried out on randomly selected images from the dataset, at various noise levels and distributions across the channels. The results of these tests can be seen in Table 5.2. The numerical average SSIM values are plotted in Figure 5.12. As the percent corruption increases, the traditional fusion algorithm tracks in a fairly linear fashion while the more advanced control and controlled autoencoder fusion algorithms retain over 90% SSIM in the face of up to 40% pixel corruption of the original image. The simplicity of the denoising autoencoder reached parity with that of the controlled fusion algorithm. The inclusion of Figures 5.13, 5.15, 5.17, 5.19, and 5.21 provides the reader a full rendering of the results of each algorithm including the proposed autoencoder based fusion method. In

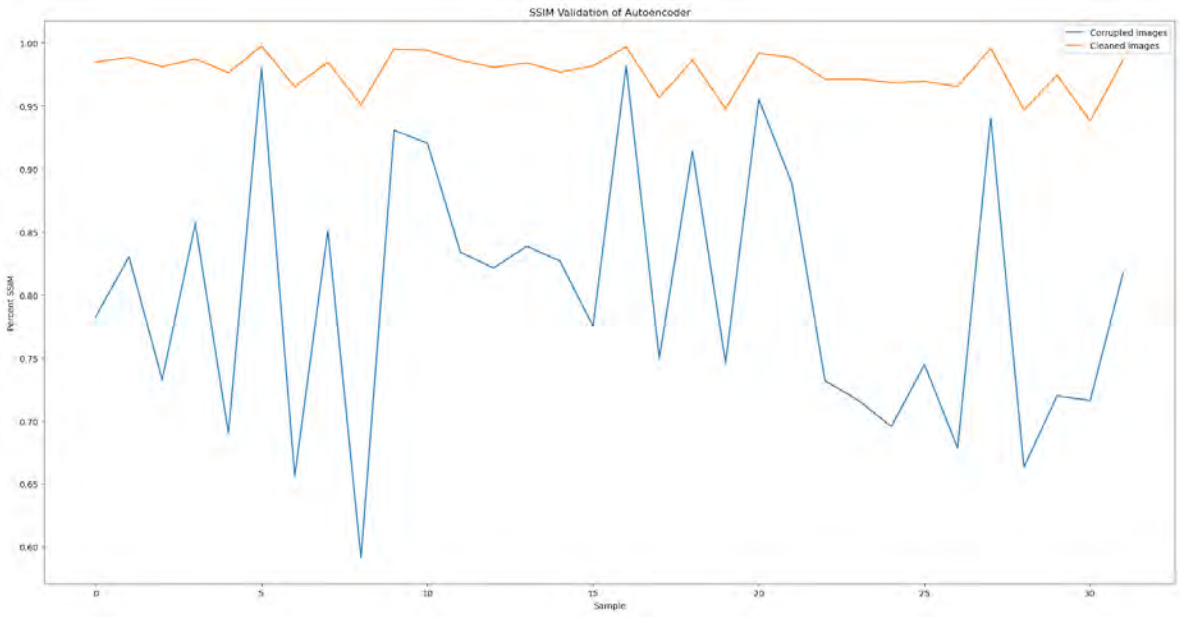


FIGURE 5.11. Comparison of the Corrupted and Output Image to the Original Sample after Autoencoder Processing Based on Structural Similarity

figures 5.14, 5.16, 5.18, 5.20, and 5.22, a SSIM heat map for each combination of images provides further efficacy for the proposed method's ability to create a valid image in the presence of corruption. The output of weighted image fusion of the original image are shown in subfigures 5.13a, 5.15a, 5.17a, 5.19a, and 5.21a at each of the percent corruption testing points, providing for visual comparison to the corrupted image fusion methods. You will notice that subfigures 5.13d, 5.15d, 5.17d, 5.19d, and 5.21d are much darker compared to subfigures 5.13e, 5.15e, 5.17e, 5.19e, 5.21e, 5.13f, 5.15f, 5.17f, 5.19f, and 5.21f.

TABLE 5.2. SSIM Table of Fusion Techniques

Percent Corruption	SSIM Image Fusion Techniques		
	Truth L2 Fusion Corrupt L2 Fusion	Truth L2 Fusion Corrupt L2 Control Fusion	Truth L2 Fusion Corrupt L2 Control Auto Fusion
5%	97.7316%	97.6116%	97.6116%
10%	94.3808%	96.9704%	96.9704%
20%	87.5132%	93.6568%	93.6568%
25%	82.8879%	96.6651%	96.6651%
40%	63.1084%	96.3707%	96.3707%
50%	59.0702%	88.4048%	88.4048%
55%	53.3545%	82.9538%	82.9538%
70%	44.7167%	66.9399%	66.9399%
90%	29.3964%	45.5480%	45.5480%

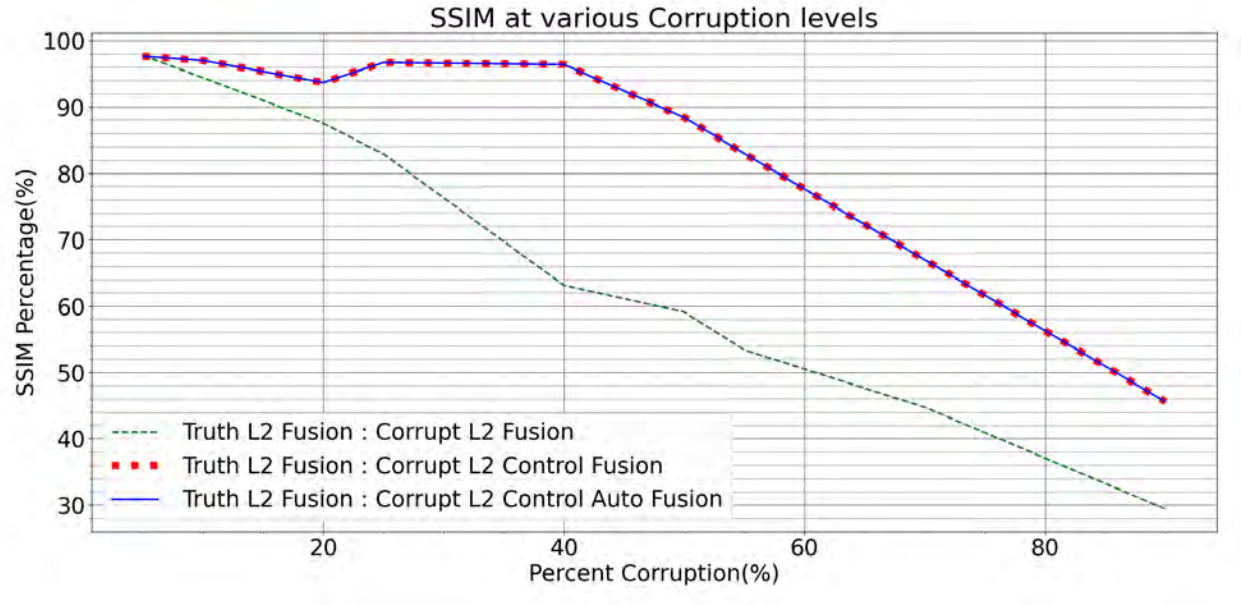
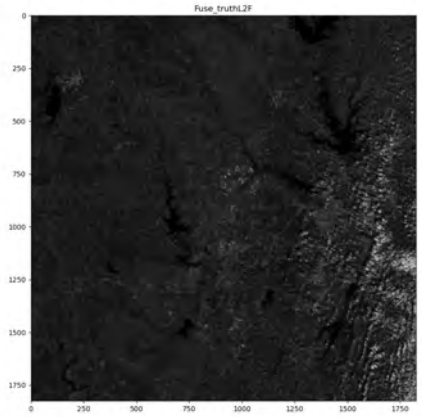
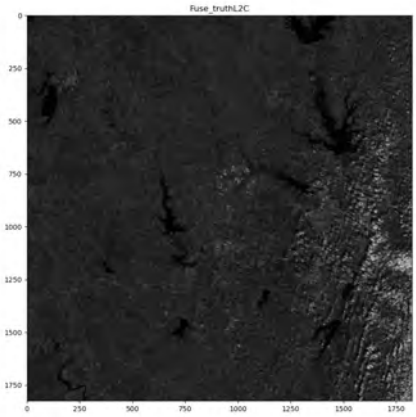


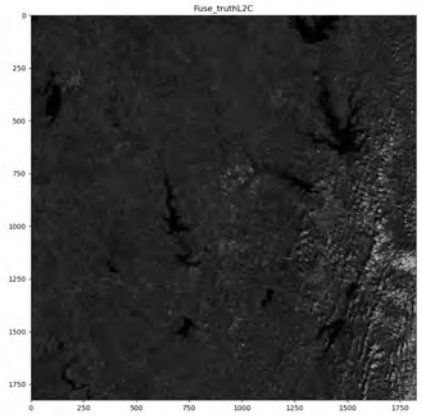
FIGURE 5.12. Average SSIM Values of Images After Full System Processing



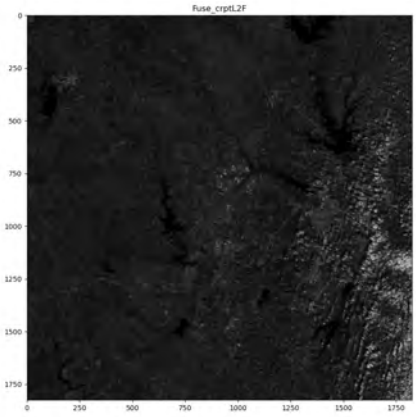
(A) Original L2 Fusion



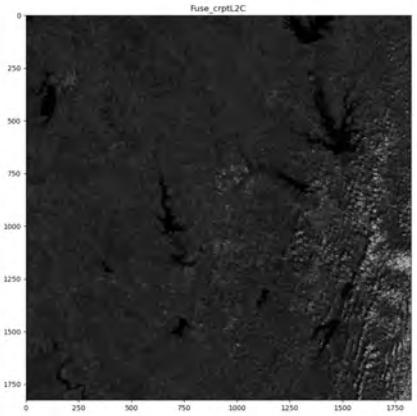
(B) Original L2 Controlled Fusion



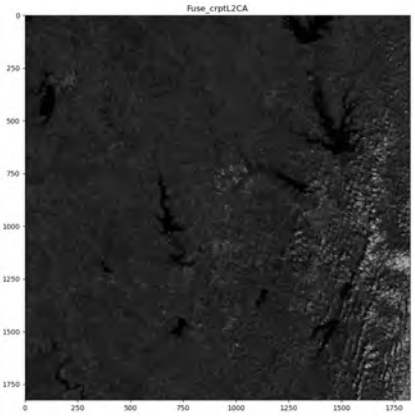
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE 5.13. Satellite Sample 1 Images Corruption 10%

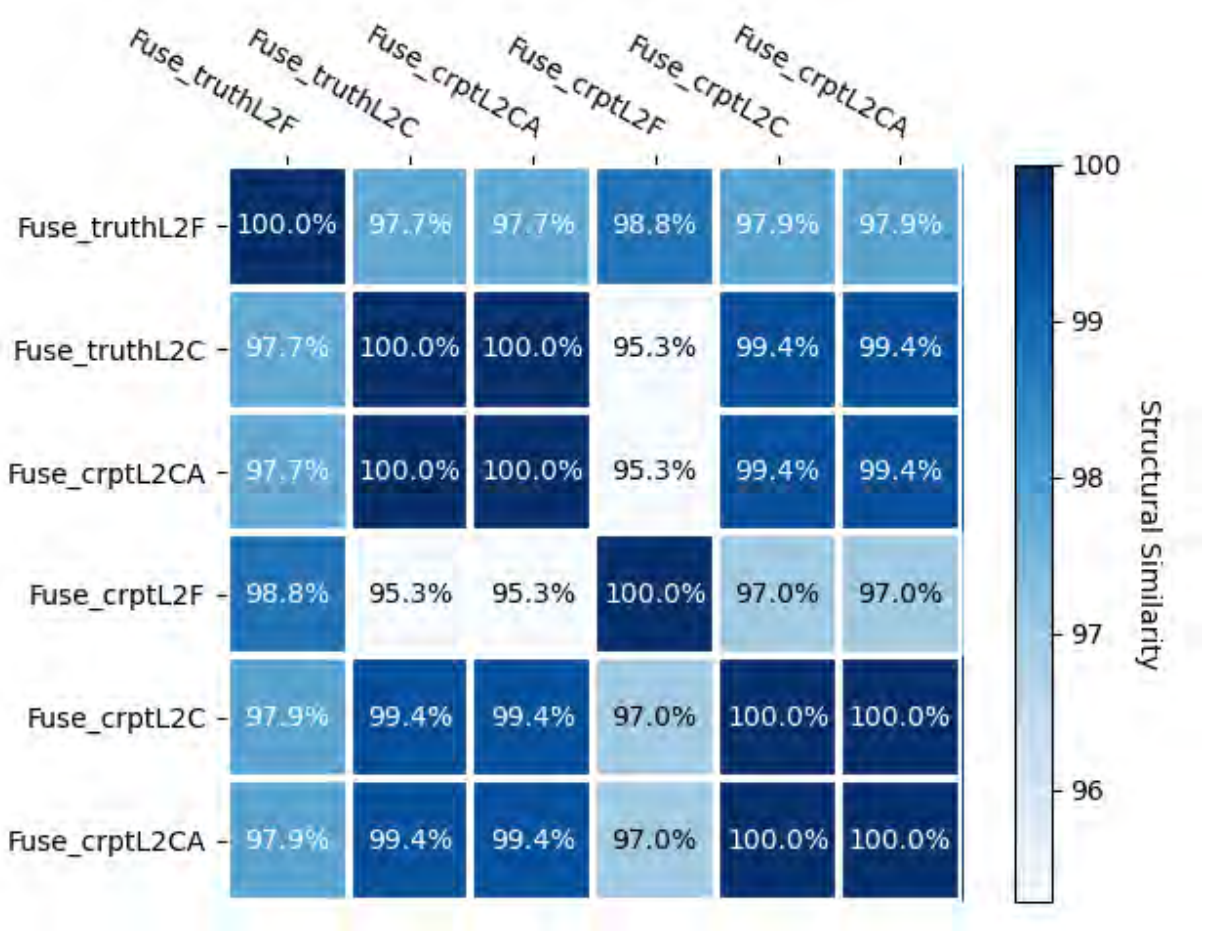
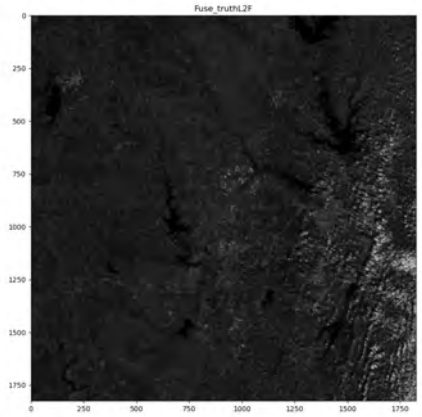


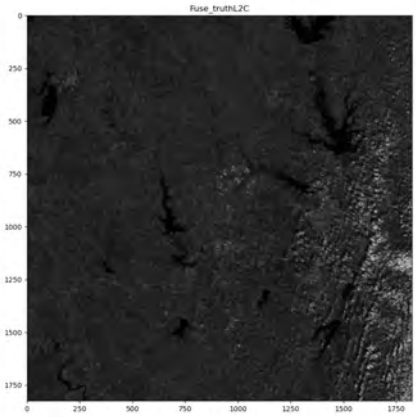
FIGURE 5.14. Satellite Sample 1 SSIM Matrix Corruption 10%

5.10. Conclusion

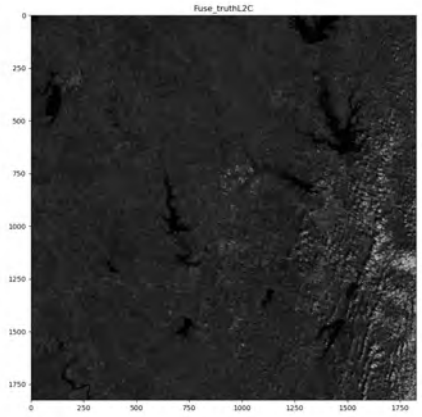
This chapter presents the implementation of a system based upon PCA techniques and neural networks that successfully fuses multi-band images in the presence of noise at a comparable level to that of controlled PCA fusion algorithms. Additional improvements can be made to the autoencoder through deeper training and development of more advanced architecture. The advances in learning systems in recent years paired with more traditional techniques has created a more dynamic system that combined with additional human interference could achieve much greater results. The analysis used blind systems to create the ground truth for the autoencoder to give an equal comparison between the systems. The inclusion of images that have been processed by human inference could lead to even better results, but were not part of the study objective. Future progress into these techniques as



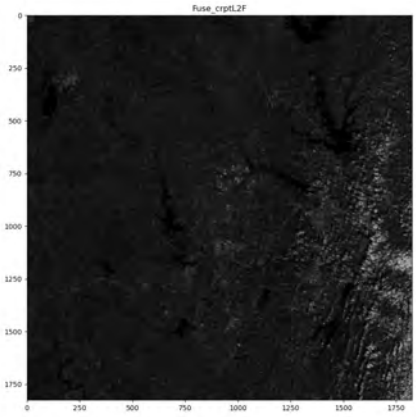
(A) Original L2 Fusion



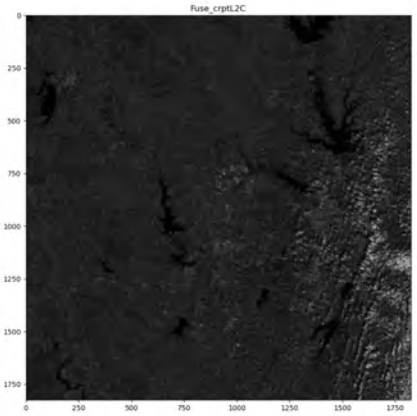
(B) Original L2 Controlled Fusion



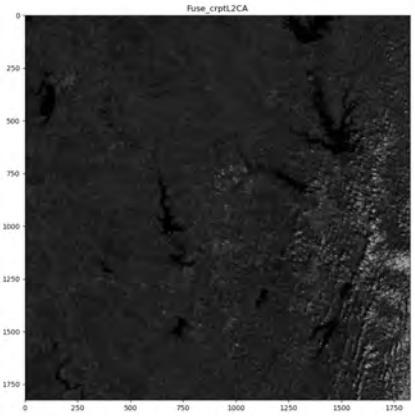
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE 5.15. Satellite Sample 1 Images Corruption 25%

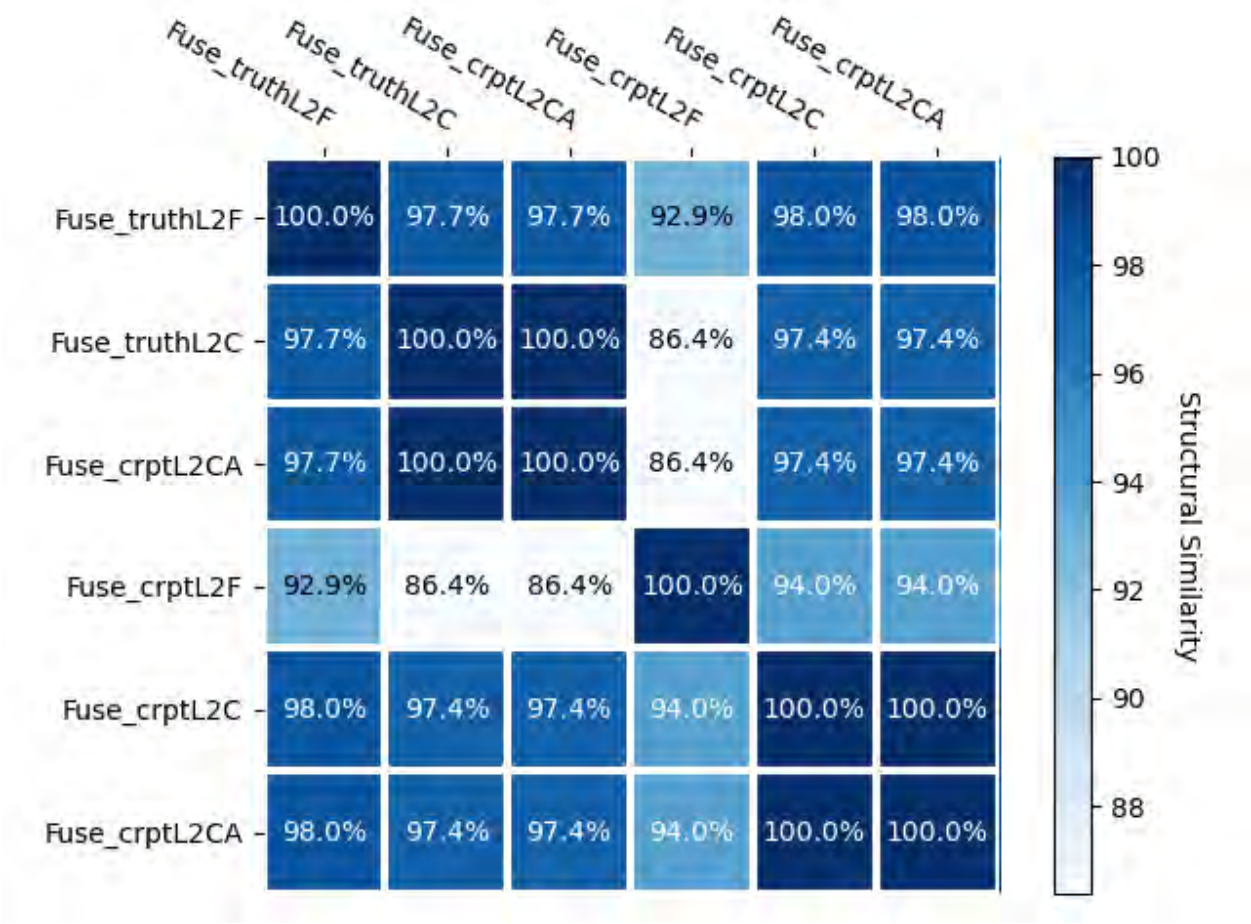
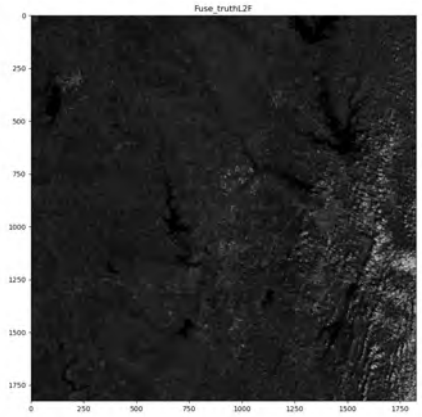
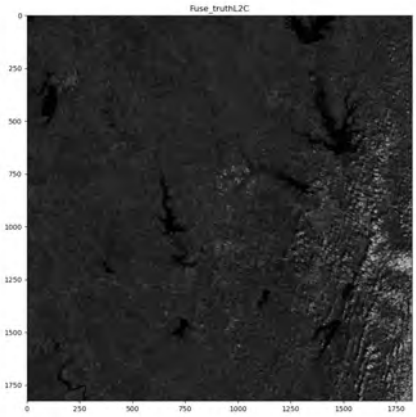


FIGURE 5.16. Satellite Sample 1 SSIM Matrix Corruption 25%

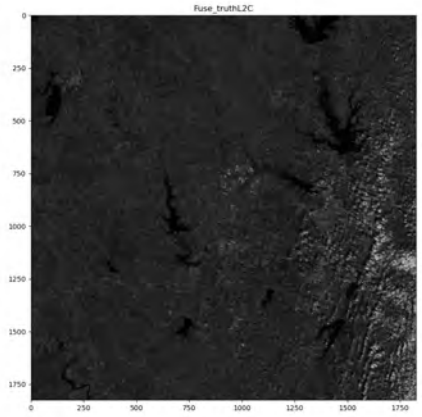
a solution to the massive amounts of computational fusion demand created on a minute by minute basis by the observational entities of the world is quite expected.



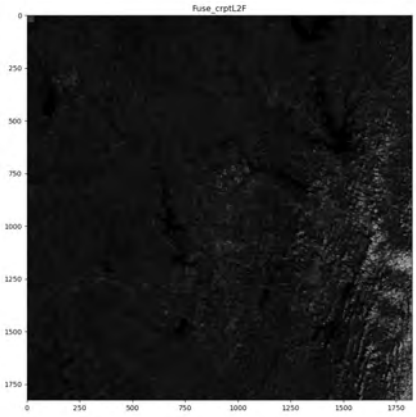
(A) Original L2 Fusion



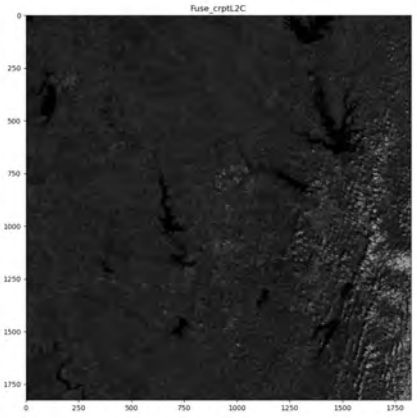
(B) Original L2 Controlled Fusion



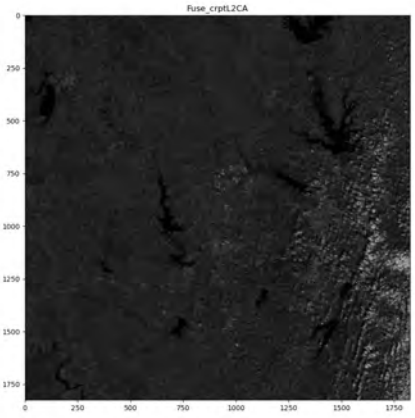
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE 5.17. Satellite Sample 1 Images Corruption 40%

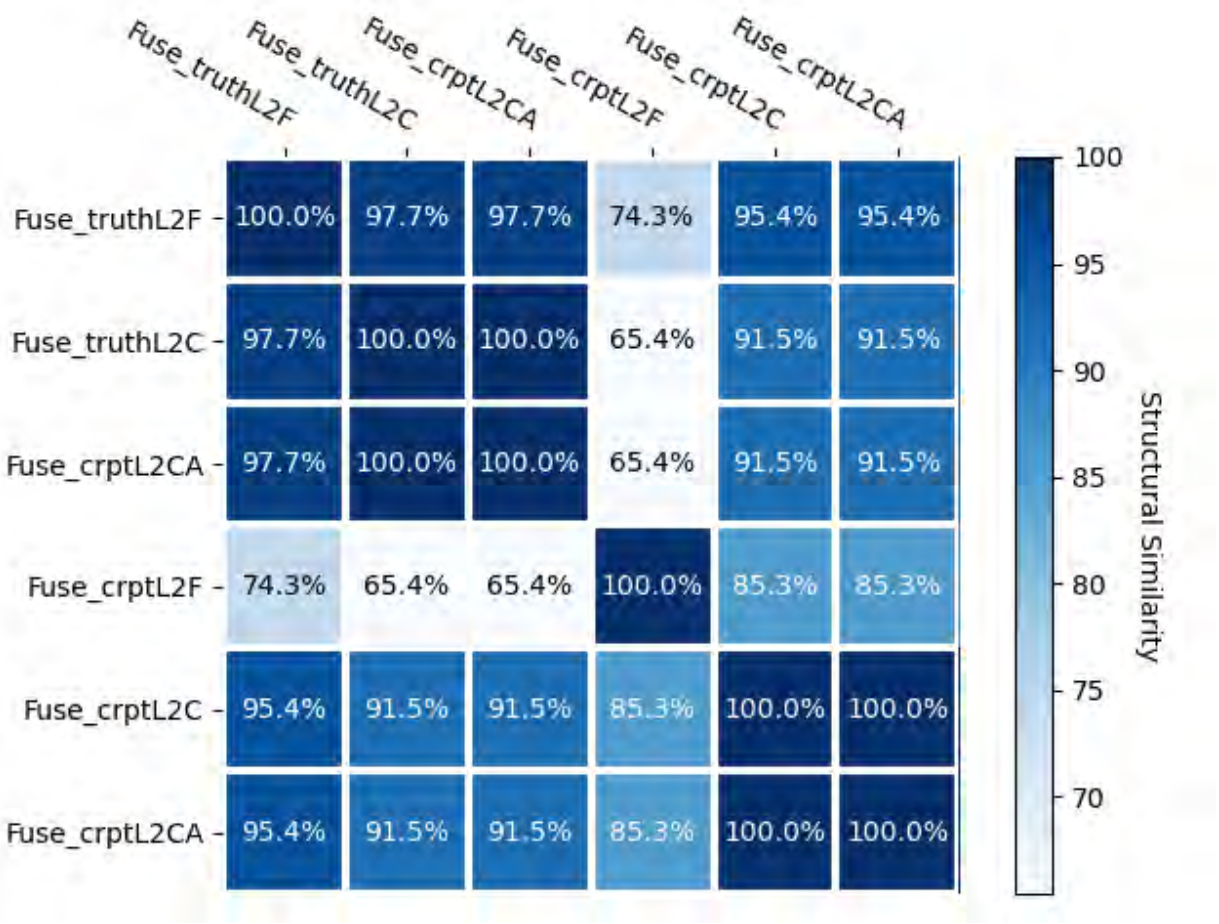
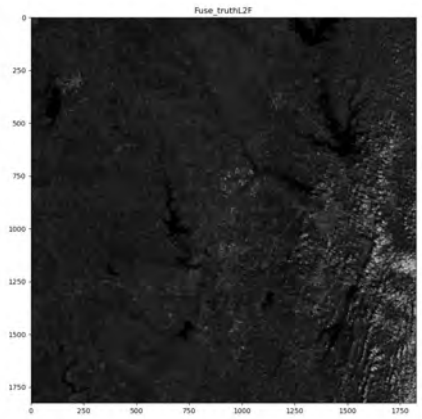
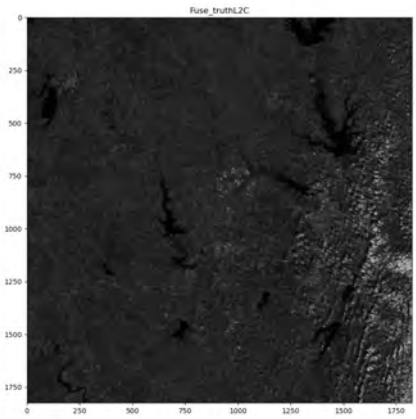


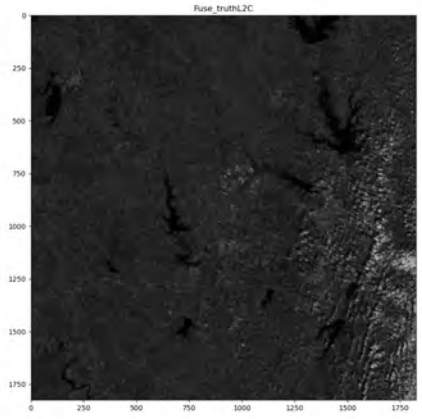
FIGURE 5.18. Satellite Sample 1 SSIM Matrix Corruption 40%



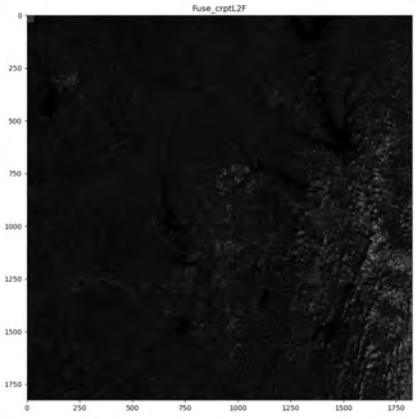
(A) Original L2 Fusion



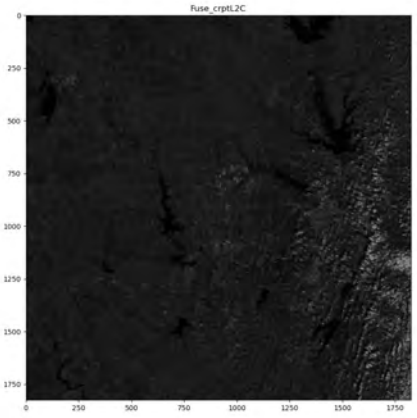
(B) Original L2 Controlled Fusion



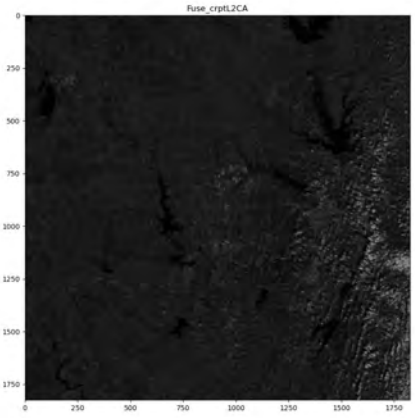
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE 5.19. Satellite Sample 1 Images Corruption 55%

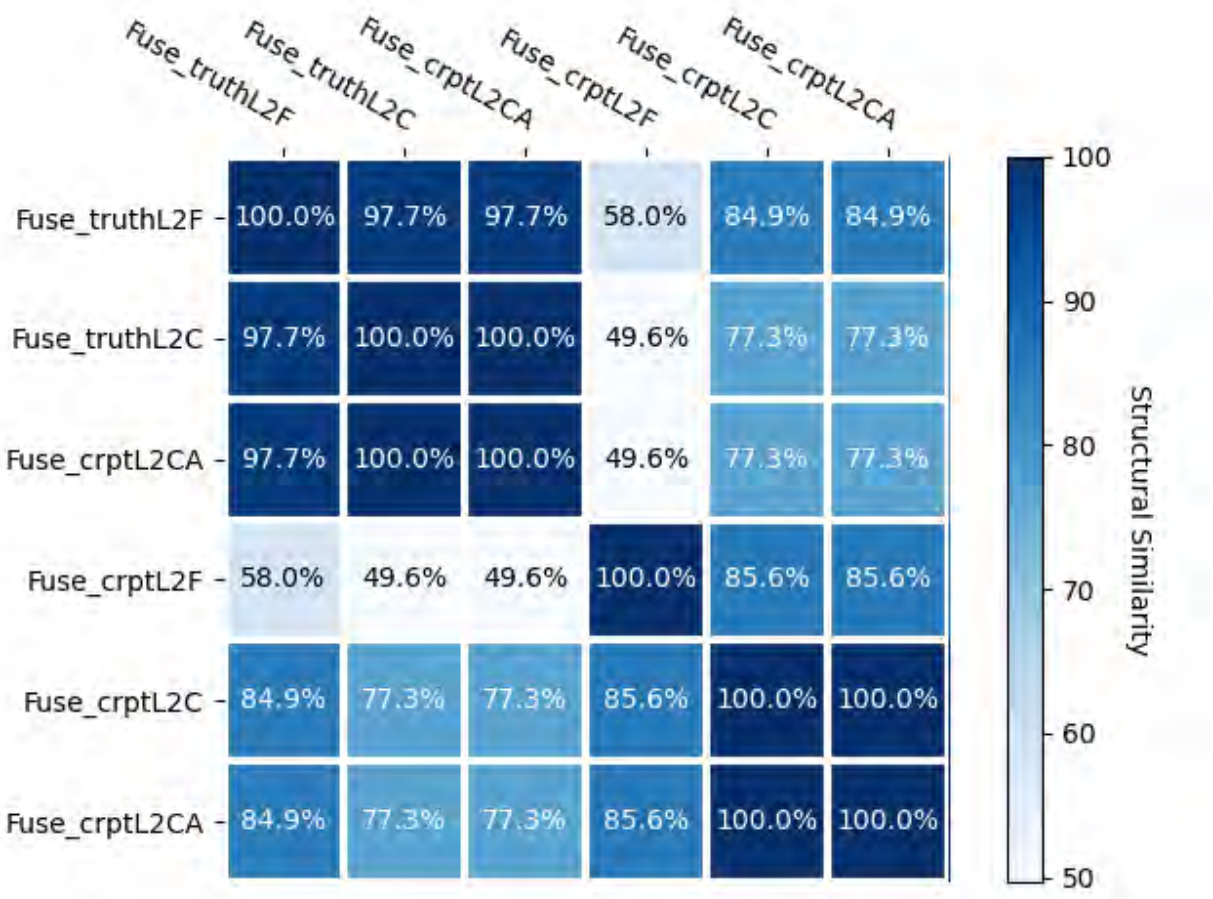
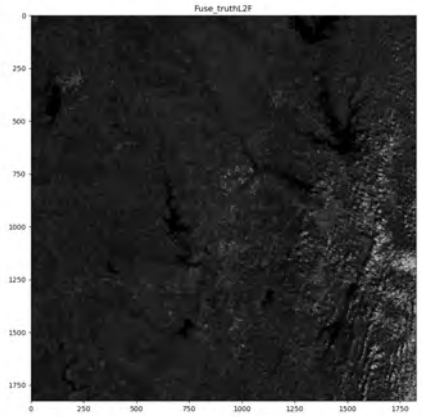
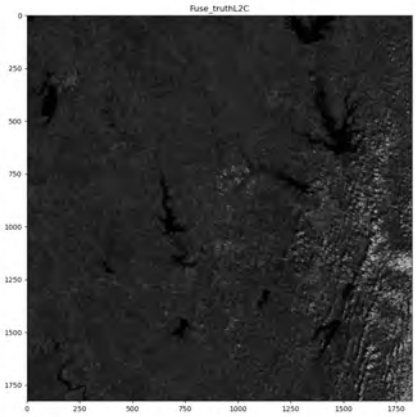


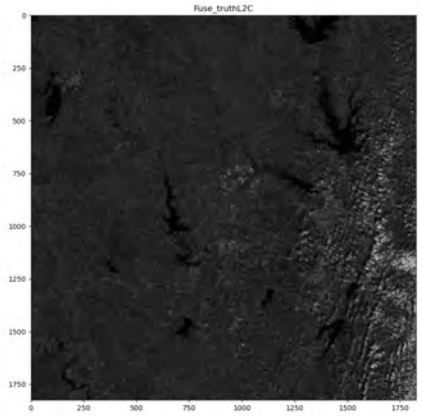
FIGURE 5.20. Satellite Sample 1 SSIM Matrix Corruption 55%



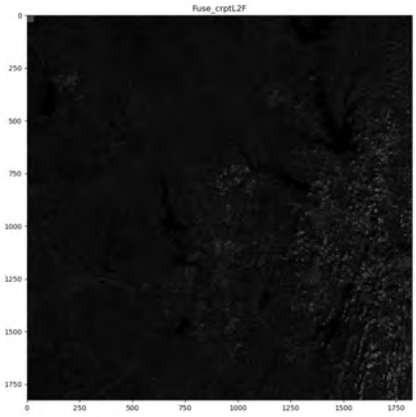
(A) Original L2 Fusion



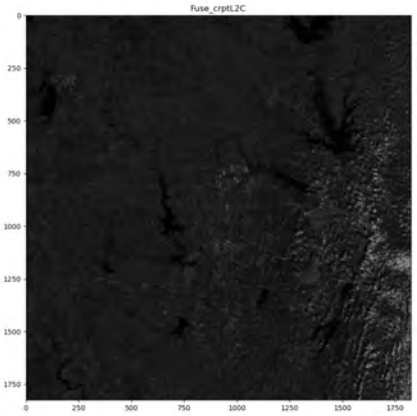
(B) Original L2 Controlled Fusion



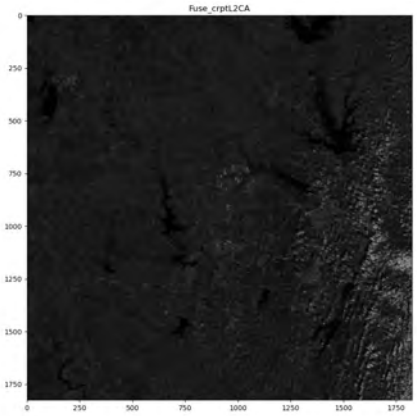
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE 5.21. Satellite Sample 1 Images Corruption 70%

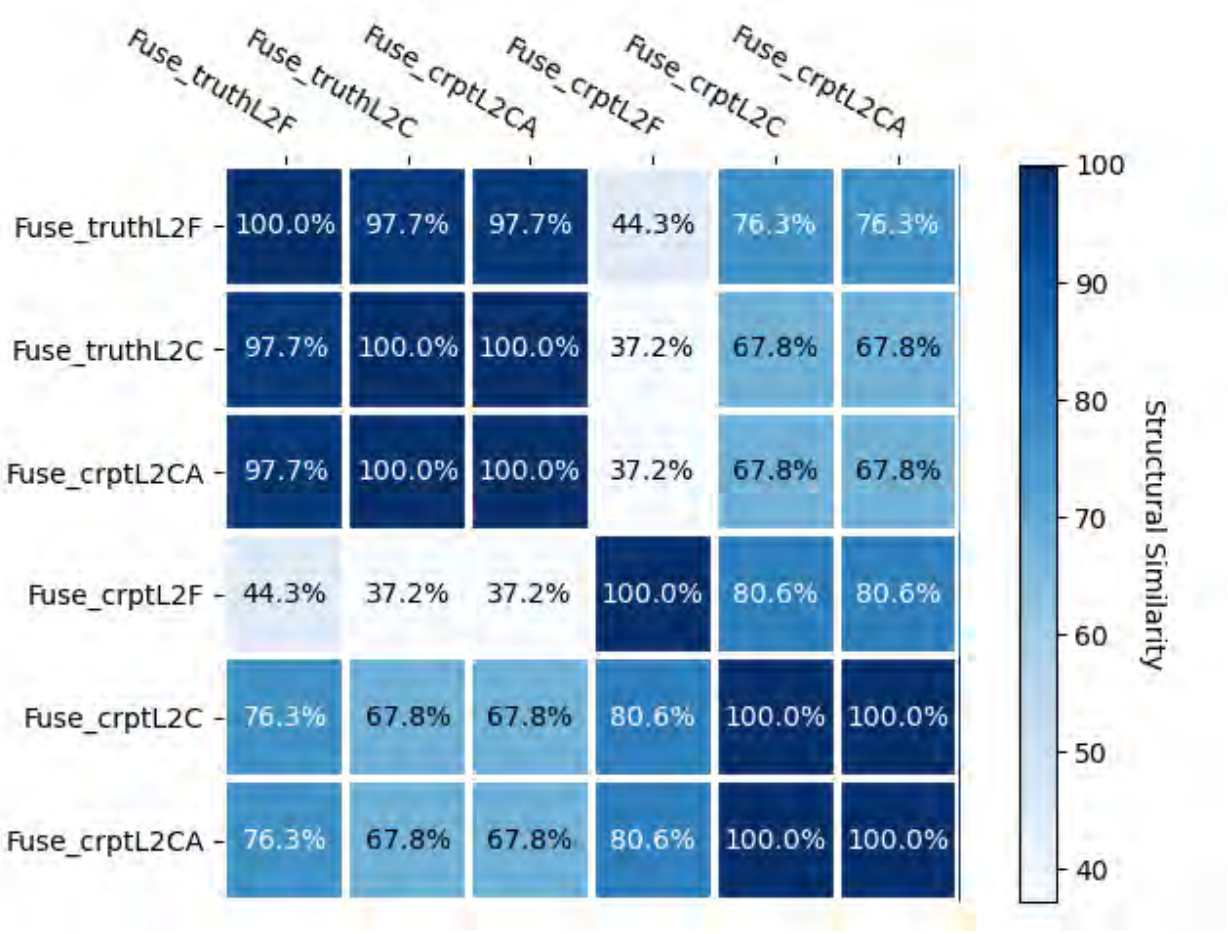


FIGURE 5.22. Satellite Sample 1 SSIM Matrix Corruption 70%

CHAPTER 6

COMPRESSED SENSING NEURAL NETWORKS APPLIED TO MEDICAL DATA

Heart disease or cardiovascular diseases (CVDs) are the number one cause of deaths annually [11, 12]. The most prolific device used to assess the heart is the electrocardiogram (ECG), comprised of sensors tuned in the acoustical range. The ECG displays the voltage versus time graph of the physiological readings. Originating as an analog device displaying on a screen or recording readings onto paper tape, it has since evolved to use analog to digital converters and digital storage media. Heath professionals have come to rely on these results, allowing for optimal diagnosis of any heart irregularities. The diagnostic analysis time is magnitudes greater than sample time and is performed by experts who have undergone years of training. This suboptimal processing technique is a small price to pay for the advantages provided by the effective results of proper identification of any areas of interest warranting further investigation. Through the spread of e-heath [70, 42] more and more data is being created that could and should be analyzed for issues. Autonomous categorization of heartbeat irregularities expedites the labor-intensive process of ECG readings, leading to a faster diagnosis in a more efficient manner. The recent advancements in wearable technology paired with artificial intelligence have allowed for a more robust heartbeat tracking and classification [9, 85]. A balance needs to be struck, taking in to account ergonomics with limited energy storage and processing power.

The MIT-BIH heartbeat database has been used as a testing set for many machine learning techniques with varied results. Sharma et al. [72] proposes a classification technique using optimal orthogonal wavelet filters for distinguishing varying heartbeats, carried out on the dataset sampled at a rate of 360 Hz. Li et al. [40] implemented a wavelet packet entropy (WPE) and random forest method for classification purposes on the same 360 Hz sampling rate resolution dataset. Kachuee et al. [29] achieve 93.4% accuracy for their analysis of categorization through their proposed deep residual convolutional neural network (CNN) implementation for data with the same data sampled at a frequency of 125 Hz.

It has been proposed in the literature that PCA techniques are not a viable solution to problems involving physiological data [78]; it has also been shown that PCA is one of the best tool for dimensionality reduction [51]. These two arguments place our goals at odds with each other, meaning a utilization of PCA on ECG readings in combination with traditional techniques and neural networks required a careful implementation of each subprocess to create an optimal solution. Highly accurate results are achieved by the application of a novel approach to medical classification using a fraction of the total data. This chapter investigates both L_1 and L_2 PCA in conjunction with a fully connected multilayered CNN to properly identify samples of interest.

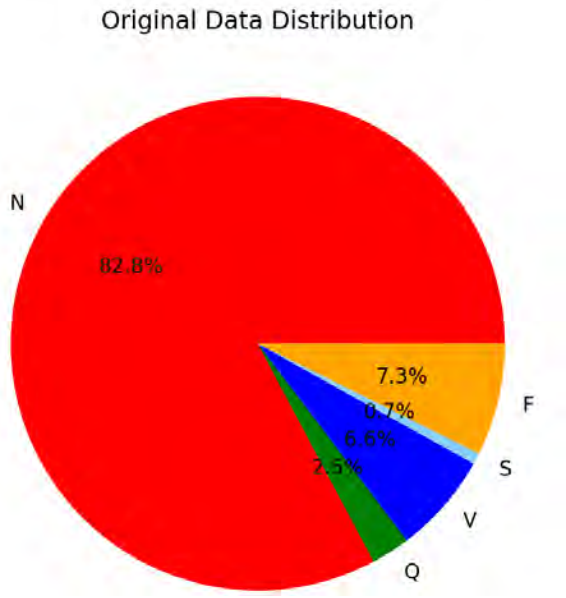
6.1. Data

The database used to carry out design and testing of the methods of PCA aided neural network analysis are that of the MIT-BIH database. The data was taken while the subject was walking and composed of forty-eight half-hour samples of two-lead ECG waveforms. The particular analog to digital conversion utilized is done at 125 Hz and 10 mV resolution for 109,446 sampled beats [54]. The expertly classified set contains 5 classes: normal (N), fusion (F), supraventricular ectopic (S), ventricular ectopic beats (V), and unclassifiable (Q) as seen in Table 6.1a. To better visualize each signal, a random sample from each category can be seen in Figure 6.2a.

The data requires balancing to mitigate skewing of outputs due to the effect of training categories unequally before input to the neural network. The normal beats category, with over 80% of the data, as shown in Figure 6.1b, would consume the majority of the training time and the remaining four categories would be under serviced. The solution is to subsample the dominant categories of the dataset, in this case their is one majority class. The prescribed solution extracts random samples from the dominant class to a count of 20,000. All categories whose sets are below the new level of 20,000 should be now oversampled up to this value to evenly redistribution the data. Each class has randomly selected samples from the original set cloned into the set until the set value is met. After this series of operations the dataset is prepared for training results.

(A) AAMIEC57 [1] Category Mapping

Label	Annotations
N	-Normal -Left/Right bundle branch block -Atrial escape -Nodal escape
S	-Atrial premature -Aberrant atrial premature -Nodal premature -Supraventricular premature
V	-Premature ventricular contraction -Ventricular escape
F	-Fusion of Ventricular and normal
Q	-Paced -Fusion of paced and normal -Unclassifiable



(B) Original Distribution

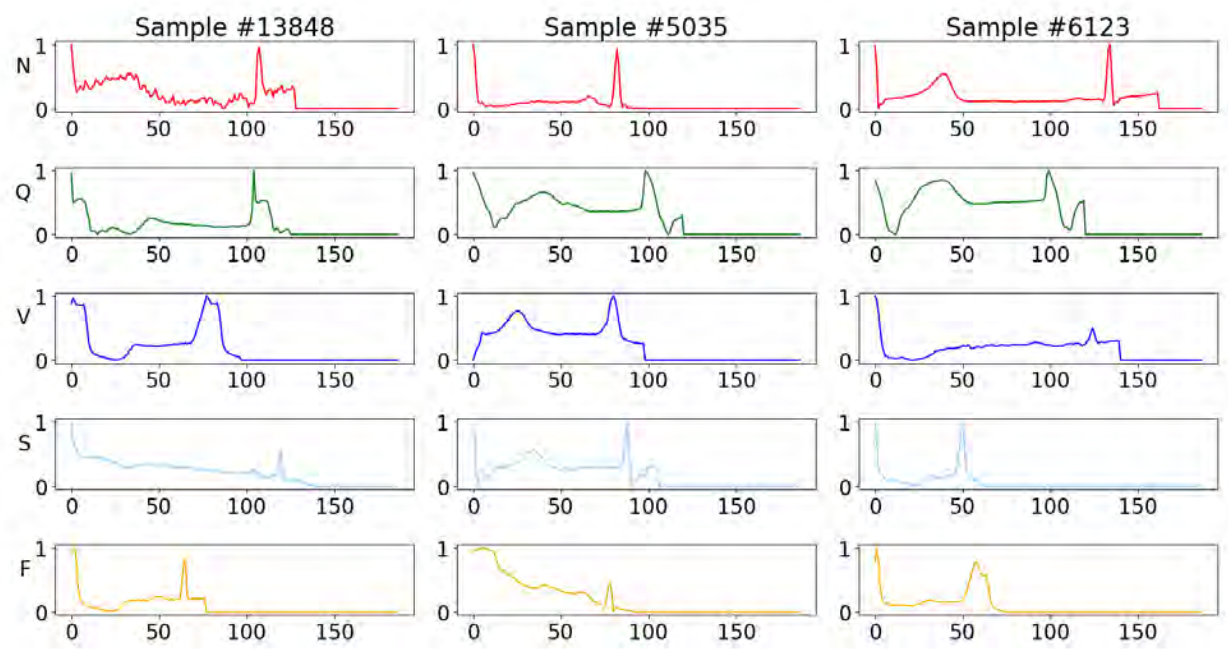
FIGURE 6.1. MIT-BIH Data

6.2. Model

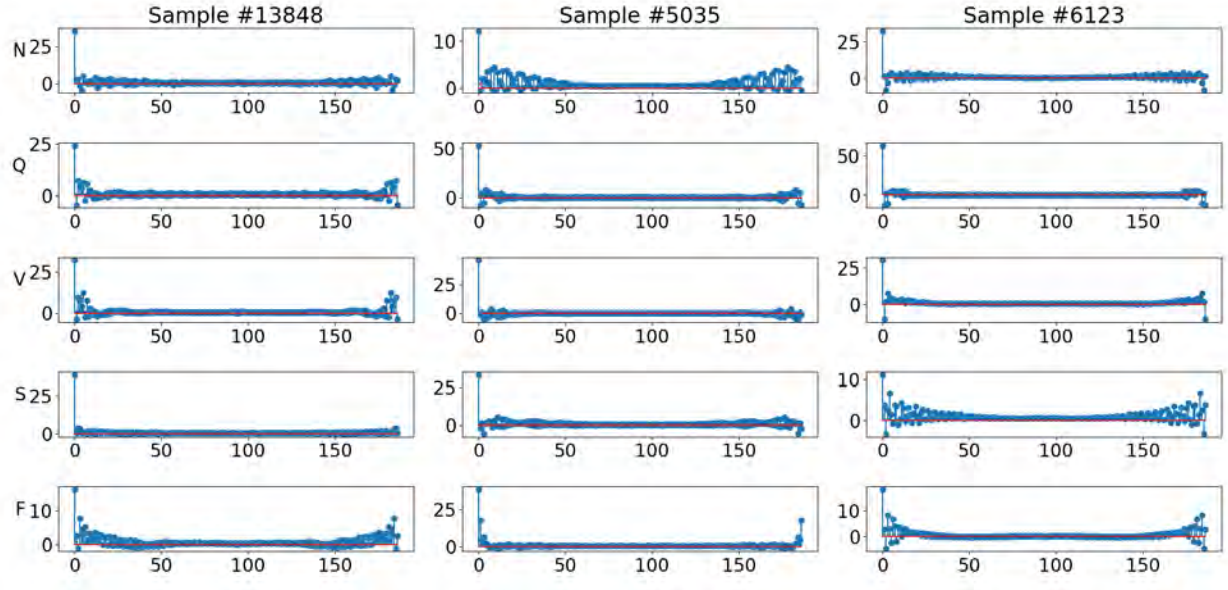
The neural network visualized in Figure 6.3, is that of the full data input as documented in [27]. It is a fully connected CNN comprised of three 1D convolutional layers, each followed by batch normalization immediately, flowing into max pooling layers that select the local maxima. These 3 layers in series feed into the dense layer that concentrates the results into five values, one for each classification label.

6.3. FFT-Fold-PCA

A novel method to preprocess the data before exposing it to the neural network is used to cut down the amount of information presented to the input of the network to less than 20% of original size. Using rank 1 principle components and the FFT in conjunction with array folding achieves a data reduction of over 80%. The data is first transformed into the frequency domain using an FFT, spreading linearly all relevant sample values. At 125Hz sampling and 188 samples, the frequency resolution is $\frac{125}{188}Hz$, the results of this transformation are displayed in figure 6.2b, inspection of these plots provides insight as to the value of the process, noting the resulting pattern created for each catagory as opposed



(A) Random Samples



(B) FFT Random Samples

FIGURE 6.2. Random Samples - Original with Corresponding FFT

to that of the time domain plots where similarities are present but not so obvious. The DC component of the resultant array is removed and the remaining 1D array is split into 6 even sub-arrays that are then joined or stacked along a new axis as displayed in Figure 6.4. This system of operations has folded the FFT of singular dimension and created a 2D

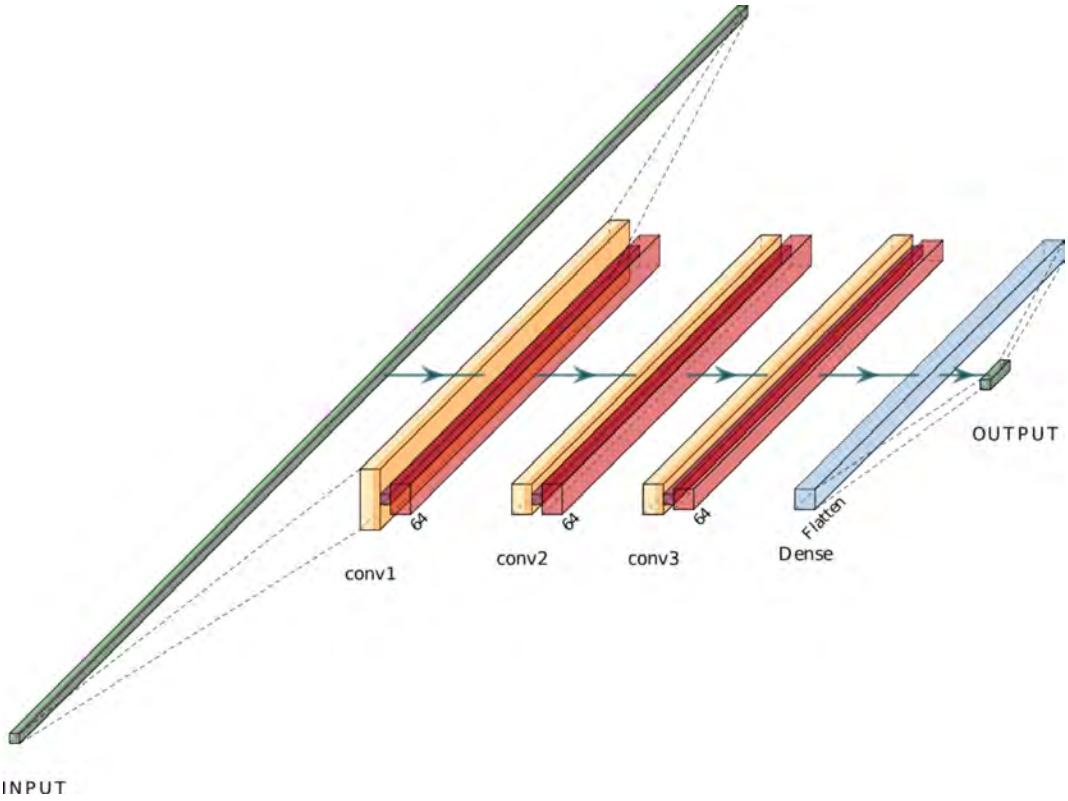


FIGURE 6.3. 1D - Convolutional Neural Network Structure Original Data

representation.

To fully grasp the power of the procedure, two parallel forks are explored to test implementations of L_1 and L_2 PCA as a modular piece of the system, directly replacing one for the other. The previously noted nature of the L_1 and L_2 principal components implies each has a purpose depending on the data characteristics and application. For these studies L_1 had been identified as the preferred method based upon the data and analysis in question.

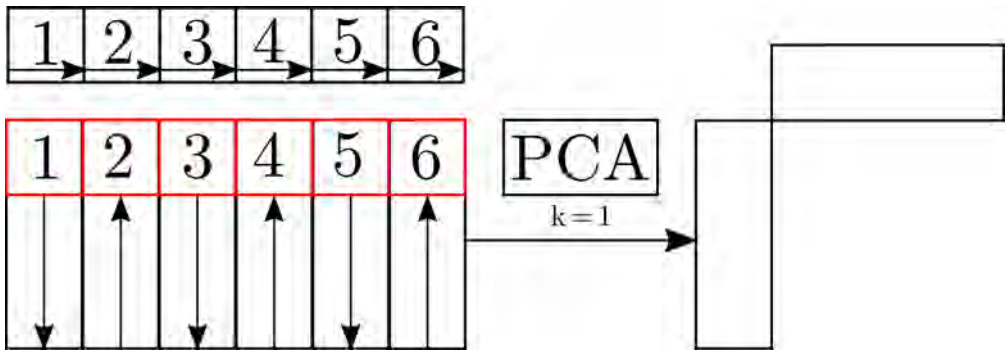


FIGURE 6.4. Folding FFT PCA procedure

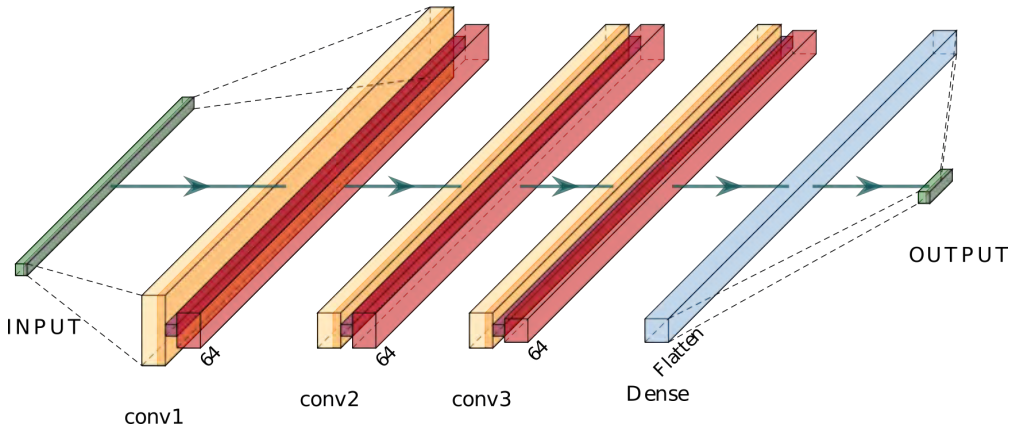


FIGURE 6.5. 1D - Convolutional Neural Network Structure PCA

To properly validate these predictions analysis was carried out with both.

The PCs are calculated using the Fast single bit flipping algorithm and the SVD method for L_1 and L_2 respectively as developed in Chapter 3. After finding the 1^{st} PCs using each of these methods, a 1D vector is created for exposure to the neural network. The implemented network is nearly identical in architecture to that proposed in Figure 6.3 with only the input size changing from 187 to 37 as displayed in Figure 6.5.

The three methods are evaluated in parallel implementations to validate the three separate networks equally. Each experimental iteration of the three distinct network is exposed to the same data for training and testing. Training and testing data sets are spread to the inputs of the original data neural network model, FFT Folded L_1 PCA neural network model, and FFT Folded L_2 PCA neural network model at the start of each experimental run of epochs, maintaining the integrity as displayed in Figure 6.6. The completion logs of the parallel experiments are stored for verification and comparative analysis.

6.4. Results

At the conclusion, testing was performed over 125 original models for each of the 3 methods under analysis. The original, L_1, and L_2 results for the testing set weighted accuracy, overall recall, overall precision, overall F1, and weighted F1 scores peak values are displayed in Table 6.1 and the average in Table 6.2.

The overall metrics are calculated based upon evenly weighting the results of the

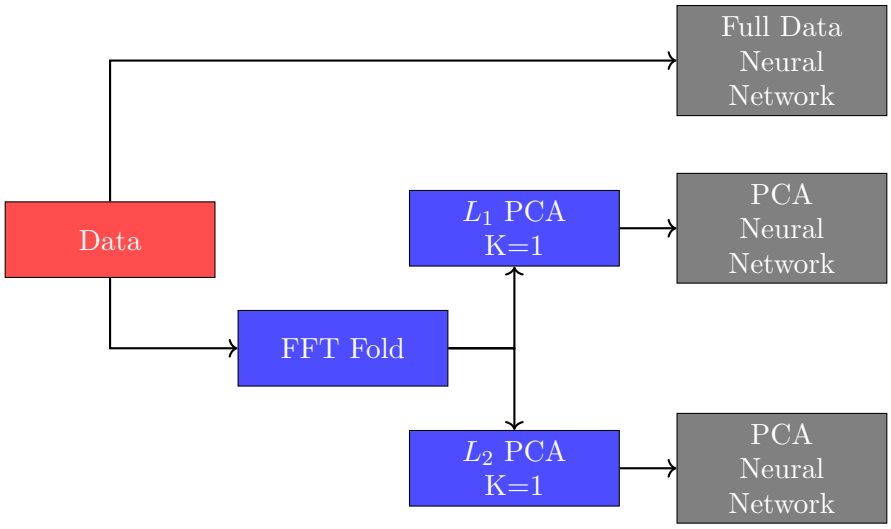


FIGURE 6.6. Global Architecture

TABLE 6.1. Peak Metrics

Metric	Original	L1	L2
Test Weighted Accuracy	98.27%	95.87%	95.46%
Test Overall Recall	93.18%	88.26%	89.40%
Test Overall Precision	90.27%	81.40%	82.63%
Test Overall F1	91.49%	84.25%	83.26%
Test Weighted F1	98.29%	96.02%	95.51%

TABLE 6.2. Average Metrics

Metric	Original	L1	L2
Test Weighted Accuracy	97.94%	94.84%	92.57%
Test Overall Recall	92.23%	86.94%	86.75%
Test Overall Precision	88.23%	77.13%	72.13%
Test Overall F1	90.05%	81.20%	77.50%
Test Weighted F1	98.00%	95.13%	93.27%

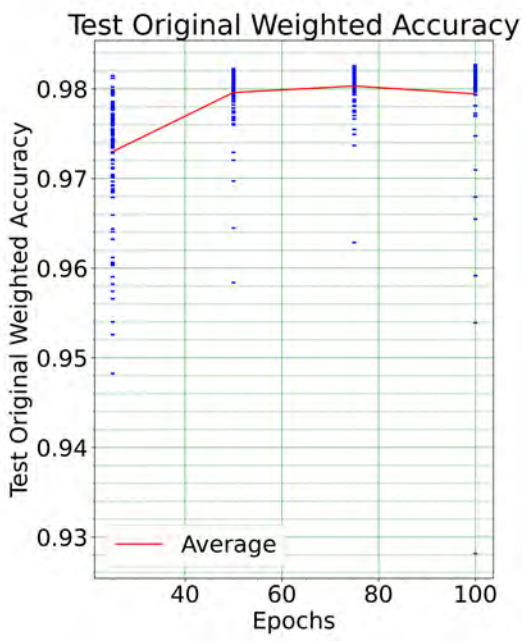
classes in comparison to that of the weighted metrics which take into account the actual distribution of each class as a percentage of the whole set. The metric of accuracy is valuable; however, it can be misleading as it does not take into account false positives and false negatives. The best metric for the network's performance is F1 score as it considers recall and precision, which both take into account the misclassifications. The inclusion of false positive and false negative information is critical to medical applications. The values over

every experiment at each epoch check point are plotted for each neural network model. Figures full data 6.7, L1 PCA 6.8, and L2 PCA 6.9 show the Accuracy and F1 scores. The full data neural network averages over 98% accuracy after 75 training epochs as shown in Figure 6.7a and over 98% F1 at the 50 epoch mark on average, plotted in Figure 6.7b. The network is processing much more information and appears to roll off at the 100 epoch results as seen by the increased range of values in comparison to the previous recorded mark displayed in Figures 6.7a and 6.7b. The results of the L_1 system plotted in 6.8 are quite astounding for a process that takes in less then 20% of the original information by size, it is still able to hit numbers in the mid 90's for both accuracy and F1. Displayed in Figure 6.8a are peak results after 100 training epochs 95.87% and an average value of 94.84%. L_1 PCA reached similar heights with the same data size with a peak accuracy of 95.46% and and average accuracy only slightly below that of L_1 at over 92% as plotted in Figure 6.9a. Figure 6.8b is the most important visualization of the results with a arching slope L_1 hits a peak score of 96.02% and average of 95.13%, the numerical values are impressive in comparison to that of the original dataset with a gap of less then 3% between them. The L_2 F1 scores again follow the L_1 with a more muted average slope, as seen in 6.9b. This is not to be discounted as they still achieve F1 scores of over 93% on average and peaking at 95.51%.

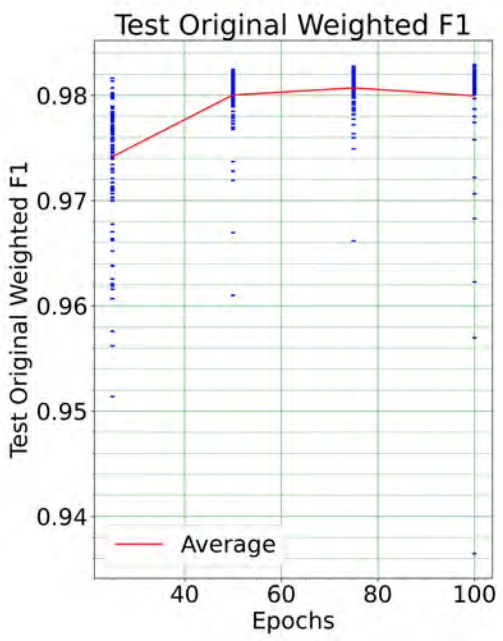
The implementation of a composite learning network requiring only 19.68% of the original information by size has led to only a slight reduction in performance metrics. The F1 metric that has become the standard for similar datasets experiencing only a 2.87% decrease when compared to the full set. PCA techniques that give a reduction in size of information have again proven to be a valuable tool, cutting down on processing time and complexity, as well as leaving the possibility open to integrate such processes in a distributed manner, multiplying the advantages.

6.5. Conclusion

The successful methods presented to classify medical data have immense value to the medical field as a future tool allowing the more efficient use of the professionals who are increasingly stretched to meet the current demands. Cardiovascular data collection has

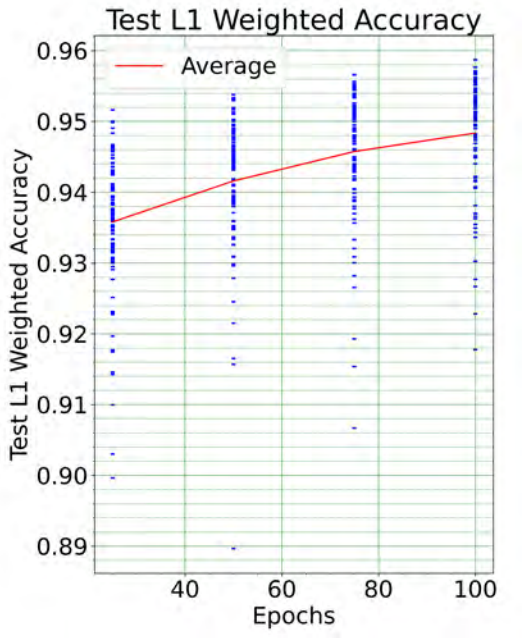


(A) Accuracy

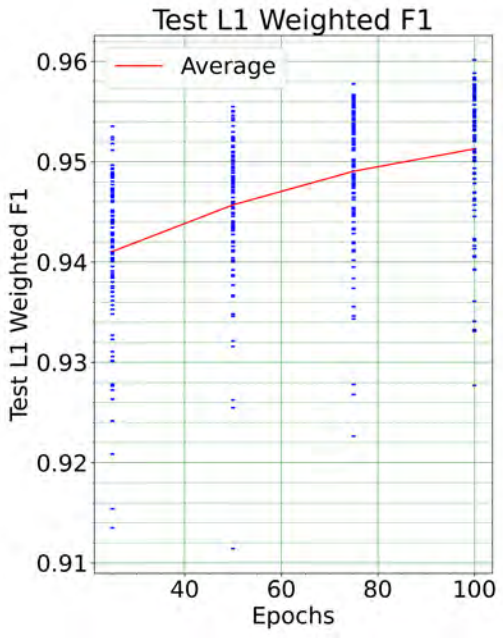


(B) F1

FIGURE 6.7. Original Model Test Results

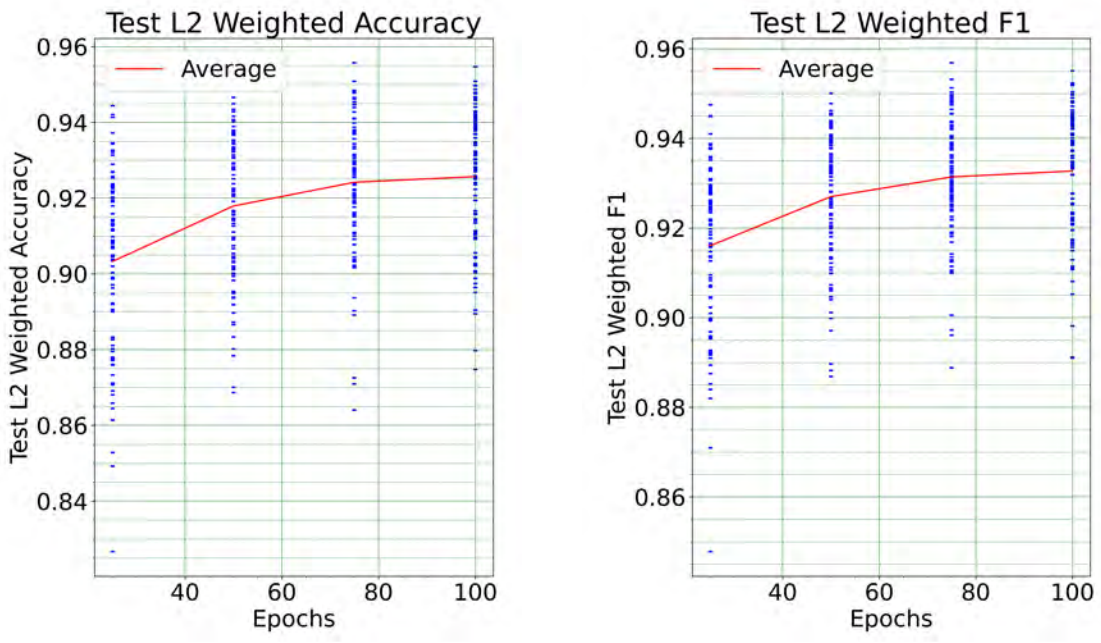


(A) Accuracy



(B) F1

FIGURE 6.8. L1 Model Test Results



(A) Accuracy (B) F1

FIGURE 6.9. L2 Model Test Results

gained great speed with the development of wearable health tracking devices moving into the consumer market, spreading technology previously segregated to the doctor's office or hospital. The reduction in size of the data at the point of input to the neural network means that the simple operations of the FFT and folding presented could be carried out on many of these devices, allowing for higher data density on storage devices and a significant reduction in transmission cost. The reduction in size of input data to the neural network also means that the same device that processes the full input data could process multiple samples in the time required to process just one. Conversely, the smaller network could be implemented on less substantial hardware, both of these options are real advantages. The processing of the complete input through the classification network hits accuracy and F1 scores that are amazing at over 98%. The accuracy and F1 scores of the L_1 method reached averages of over 95%, it is a reasonable to consider these figures as comparable rates of identification to that of the top cardiologists in the world. Even the numbers of the L_2 method are respectable in the low 90's.

CHAPTER 7

CONCLUSION

This thesis has covered two focus areas, computer vision applied to video footage and machine learning algorithms and applications to meteorological data error detection, multi spectral images fusion and ECG medical data classification.

The development of a novel computer vision technique to identify overlays contained in video footage such as that contained in sporting events was developed as a tool to optimize the performance of other video analysis. A autonomous masking algorithm using Otsu's method paired with Canny edge detection builds the window filter that successfully finds the overlays of the frames.

A good amount of machine learning is carried out by the direct and indirect application of principal component analysis. The algorithms used to compute the L_1 or L_2 principle components are of particular interest. The current state of the art algorithms are successfully developed or utilized respectively in Python to provide guidance to their implementation. Application of these algorithms to remove erroneous data contained in meteorological records leads to the successful removal of errors. The removal of this data from the set can be used to build a more accurate predictive model.

The methods developed utilizing PCA into a solution for the fusion of multispectral satellite images containing noise achieves a higher performance rate than that of the currently employed methods at equivalent levels of corruption. The traditionally controlled signal processing method eliminates the pixels in error and the autoencoder integrated method corrects the pixels in error, both of these methods achieve parity as enacted in this research.

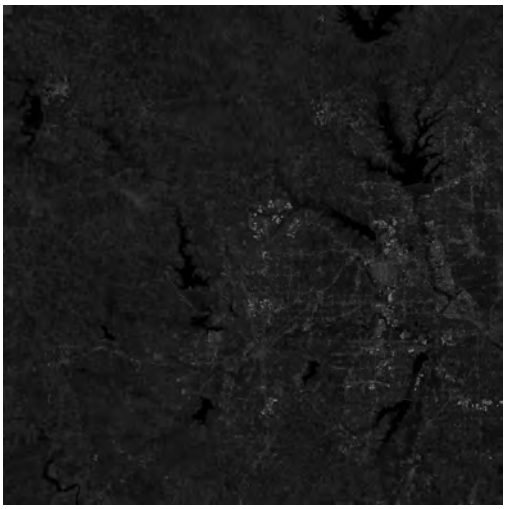
Medical data classification of PCA sparsed data through a three layer convolutional neural network experiences only a slight reduction in performance metric of only a few percent while maintaining less then 20% of the original data size. A solution to deliver results to a great number of people at reduced storage and transmission cost while not sacrificing the quality of the results, the efficacy of this method presents a strong argument for further

integration and exploration.

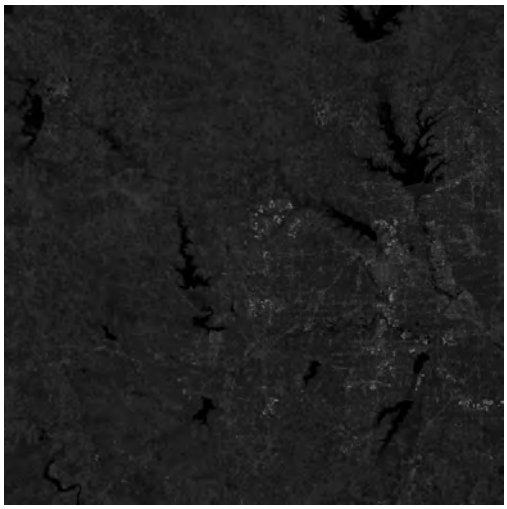
These solutions are each individually the result of the thoughtful application of the engineering process to identified problems.

APPENDIX

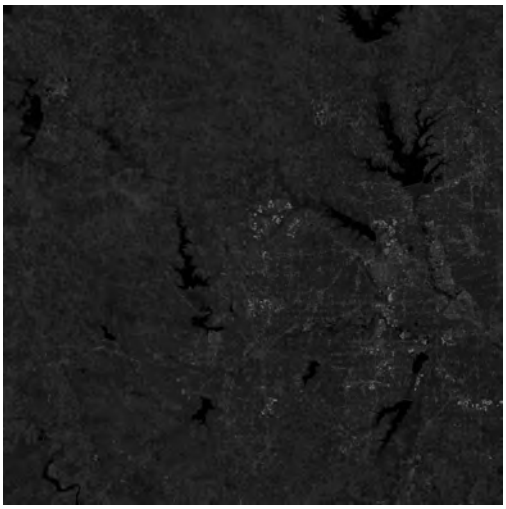
IMAGE FUSION PLOTS



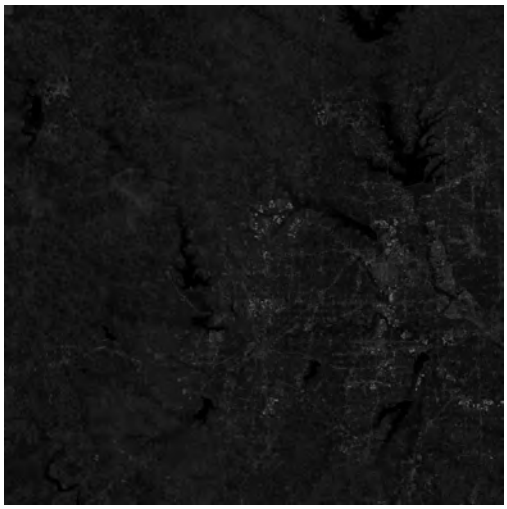
(A) Original L2 Fusion



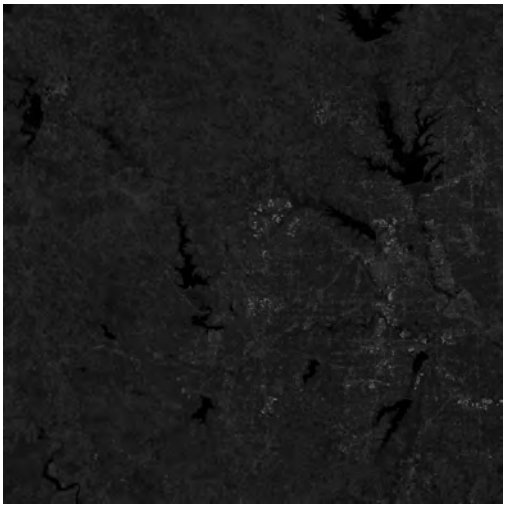
(B) Original L2 Controlled Fusion



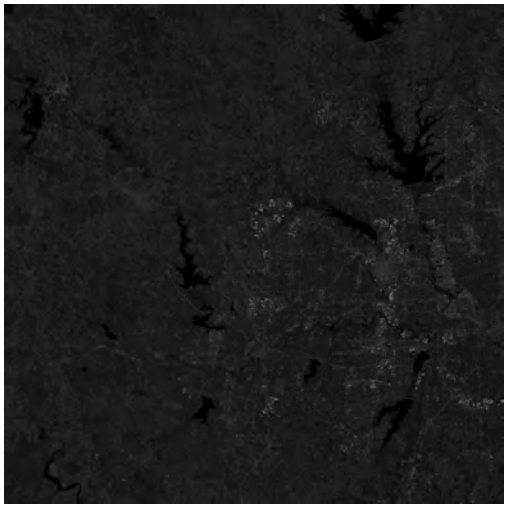
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE A.1. Satellite Sample 2 Images Corruption 10%

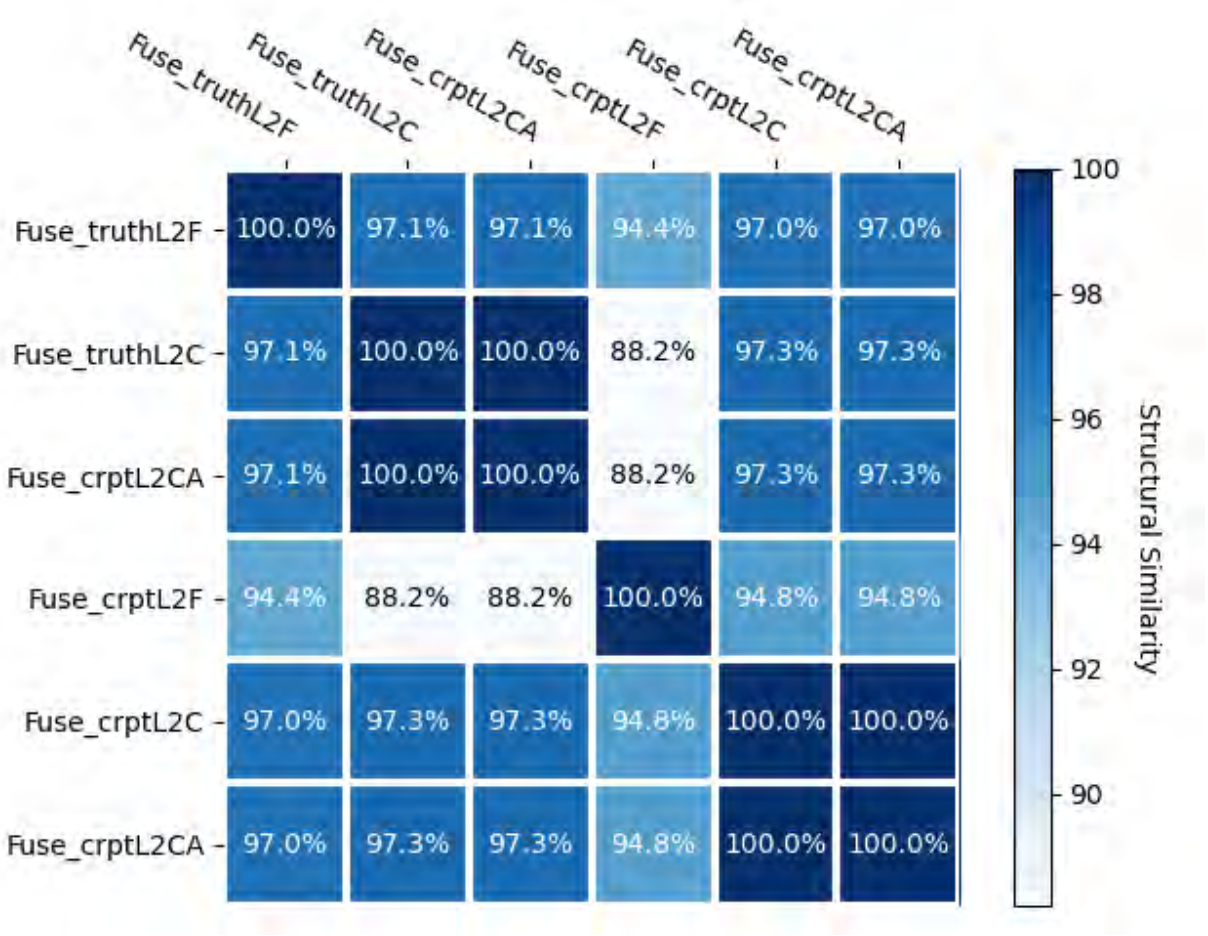
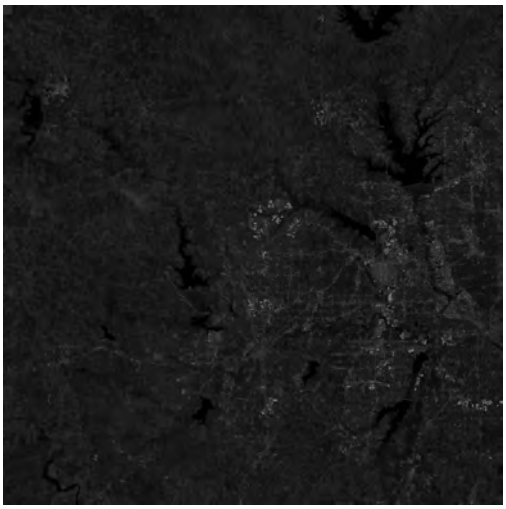
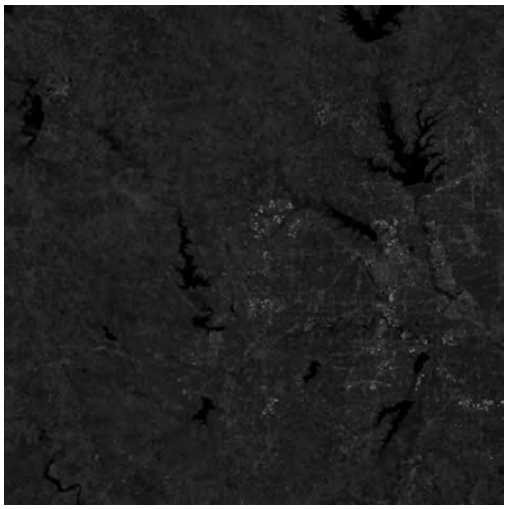


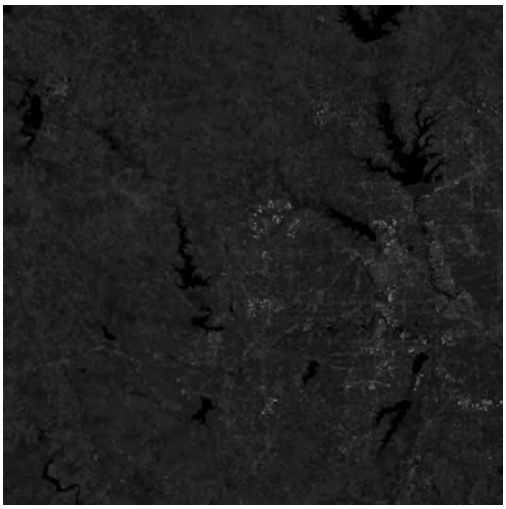
FIGURE A.2. Satellite Sample 2 SSIM Matrix Corruption 10%



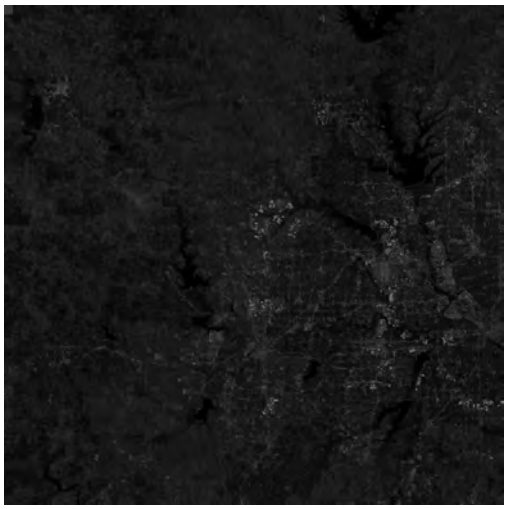
(A) Original L2 Fusion



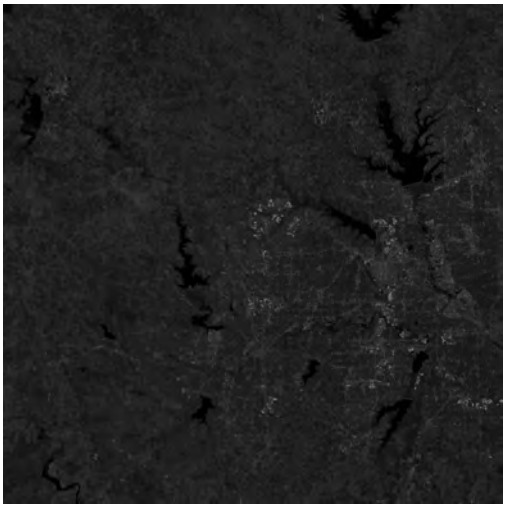
(B) Original L2 Controlled Fusion



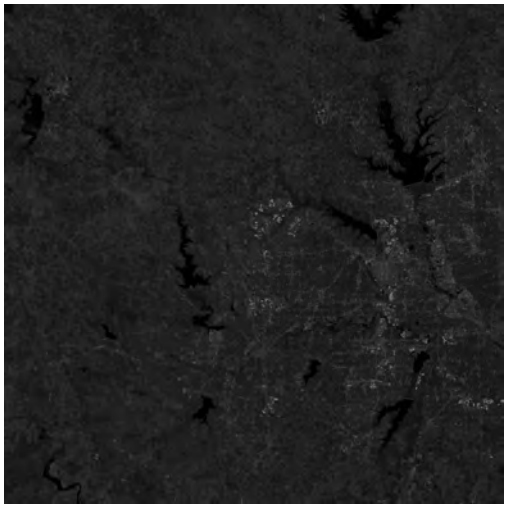
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE A.3. Satellite Sample 2 Images Corruption 20%

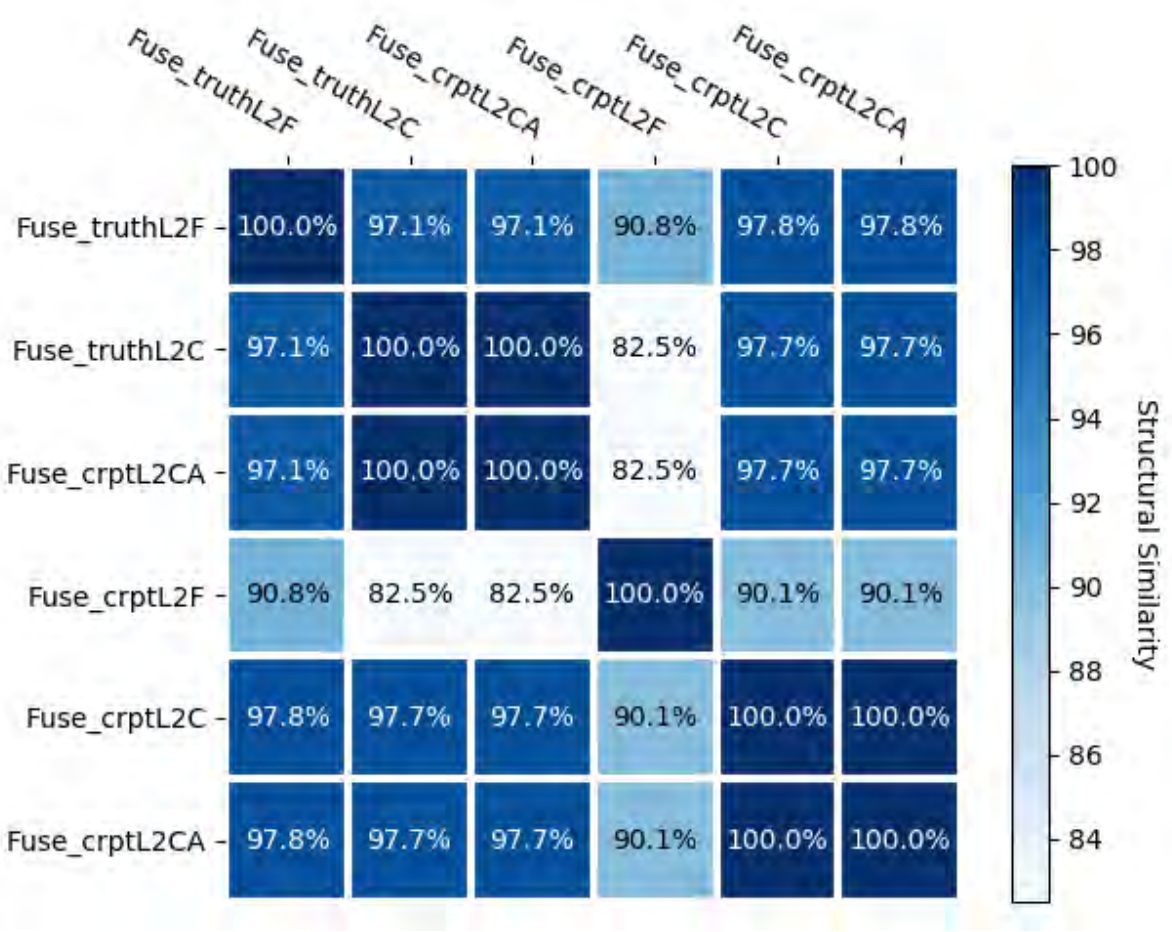
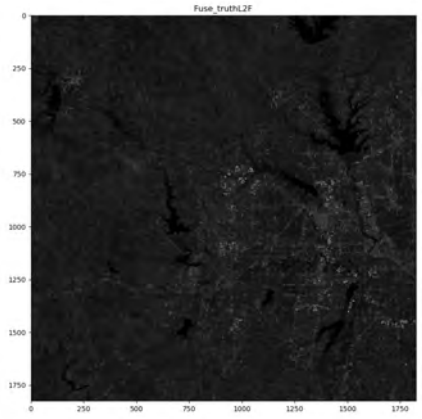
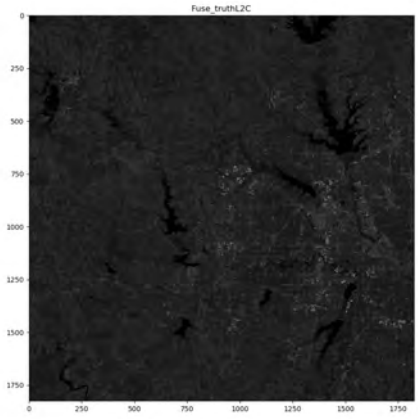


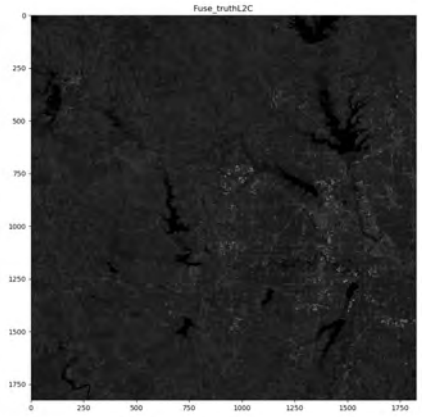
FIGURE A.4. Satellite Sample 2 SSIM Matrix Corruption 20%



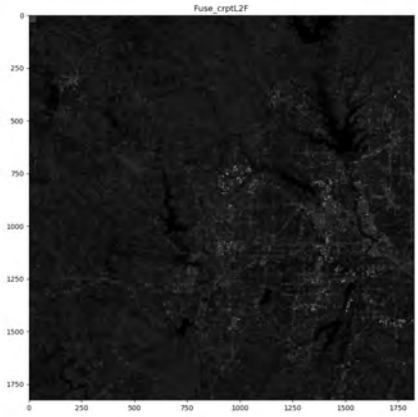
(A) Original L2 Fusion



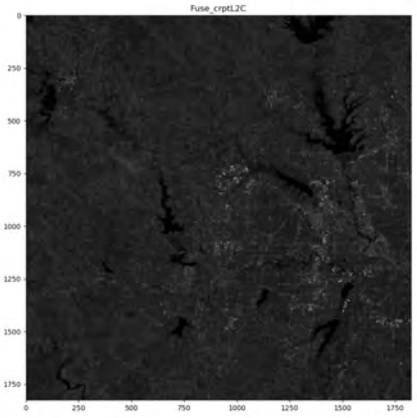
(B) Original L2 Controlled Fusion



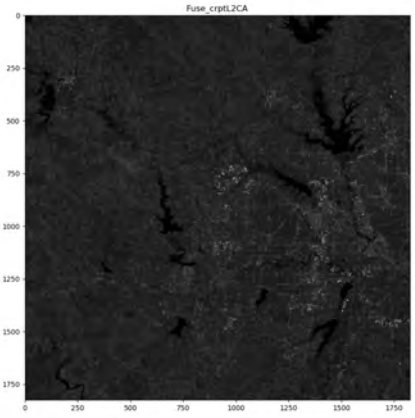
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE A.5. Satellite Sample 2 Images Corruption 25%

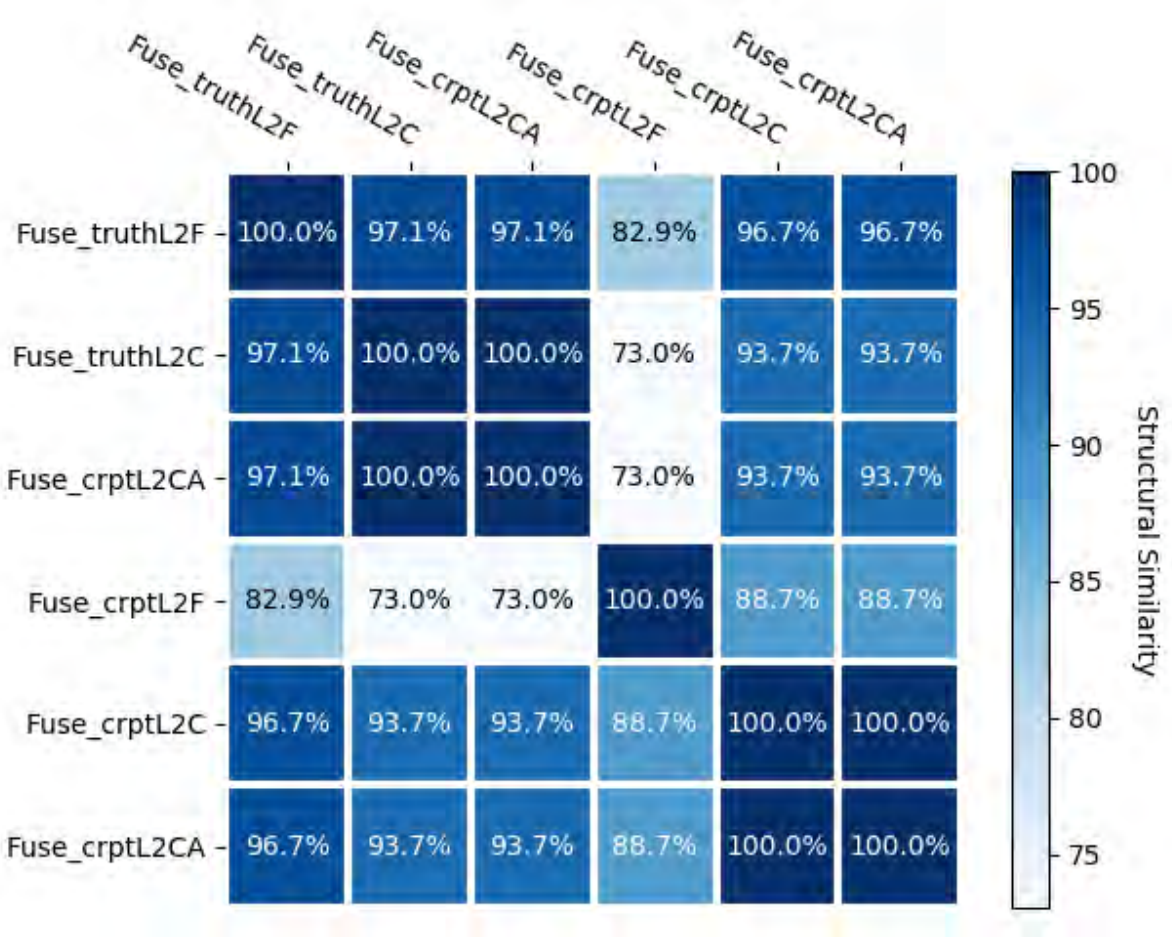
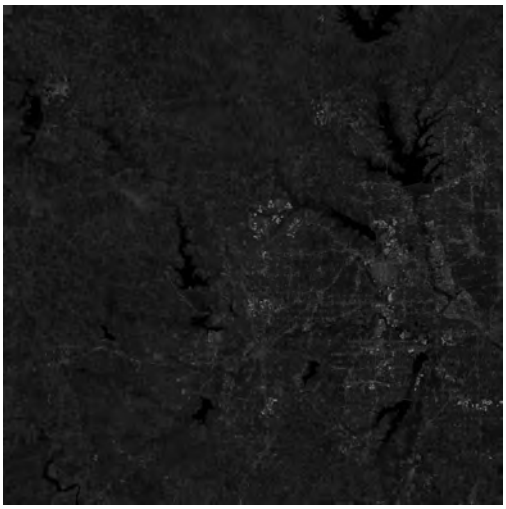
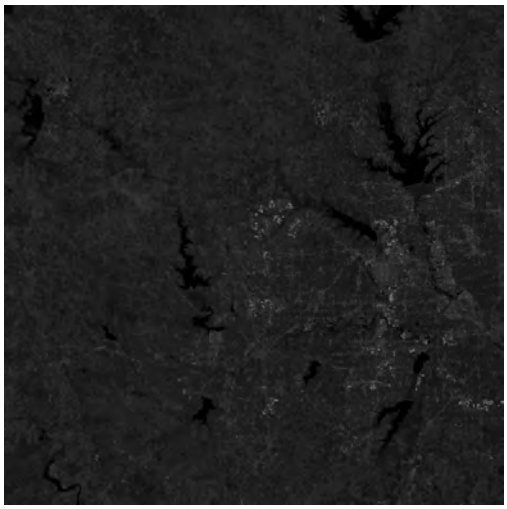


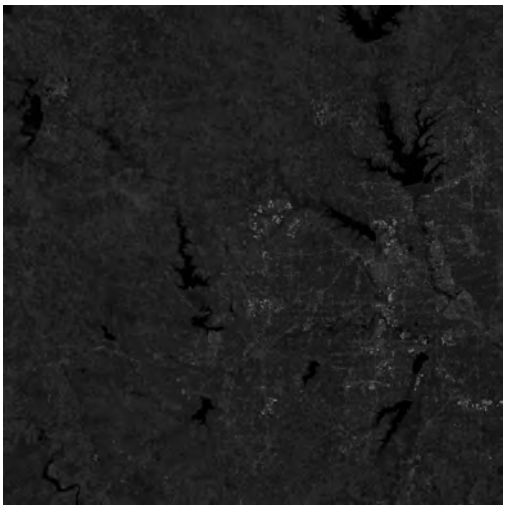
FIGURE A.6. Satellite Sample 2 SSIM Matrix Corruption 25%



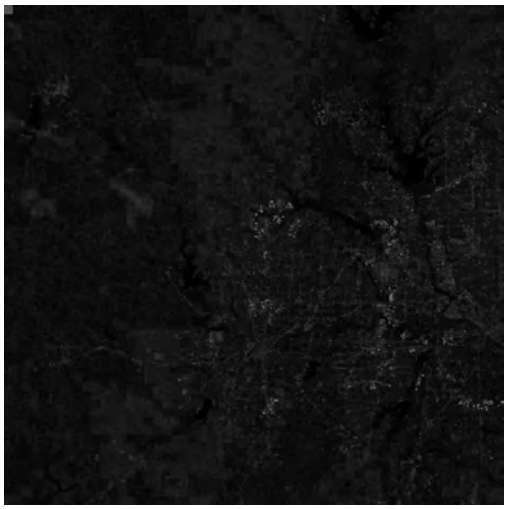
(A) Original L2 Fusion



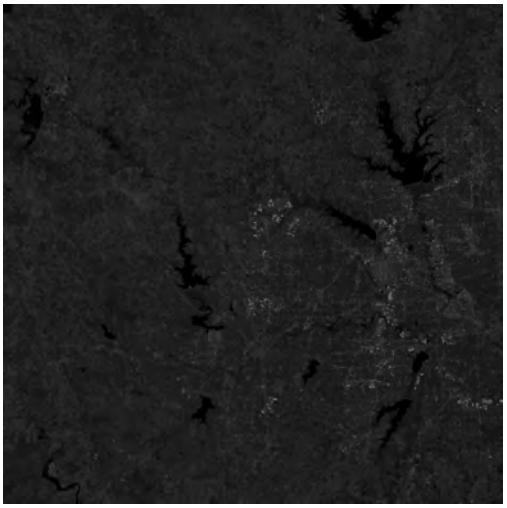
(B) Original L2 Controlled Fusion



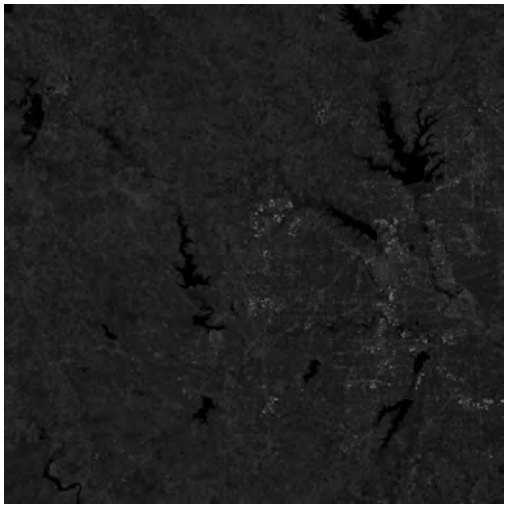
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE A.7. Satellite Sample 2 Images Corruption 40%

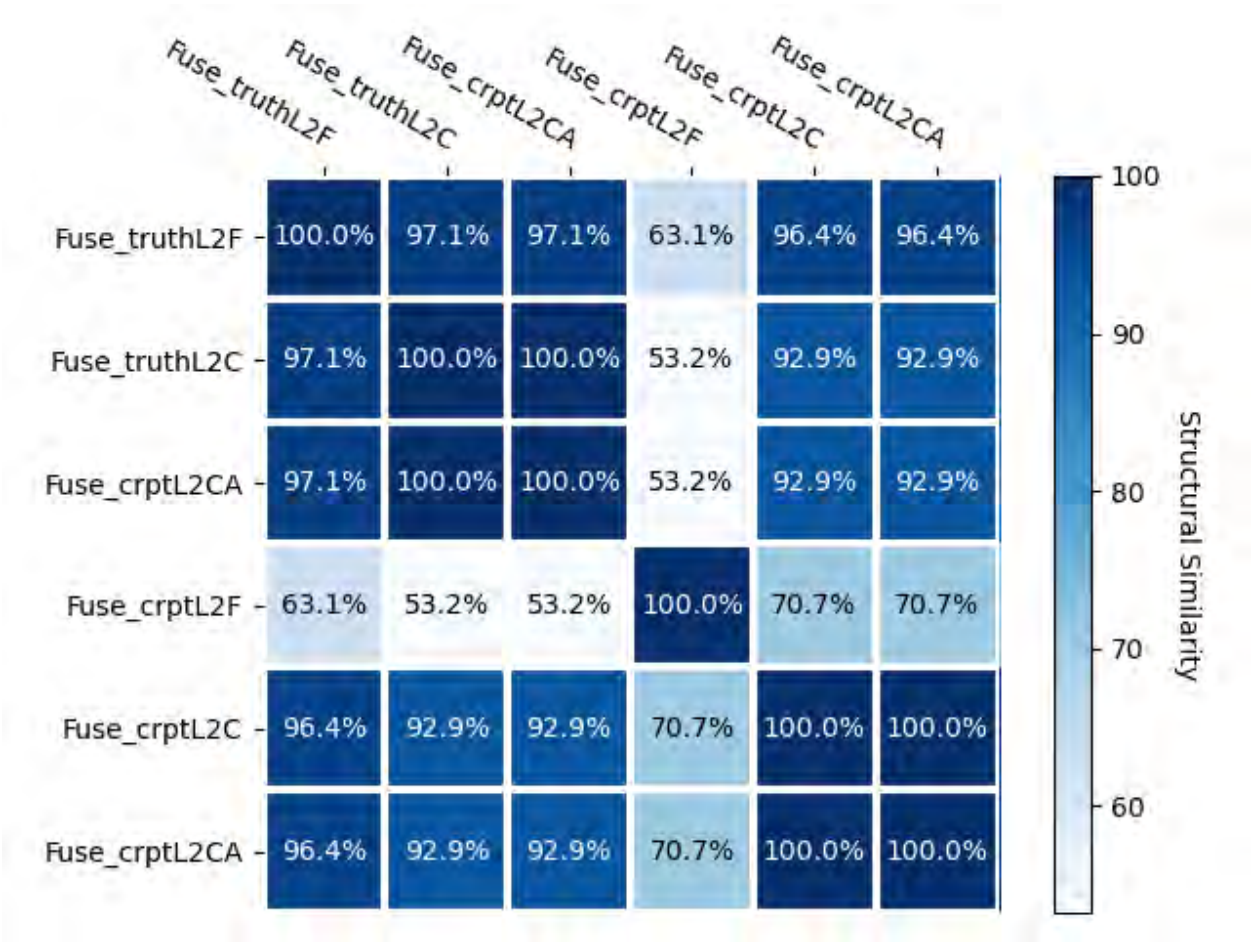
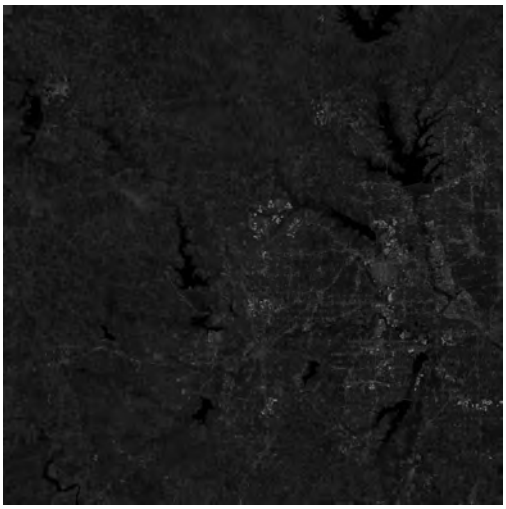
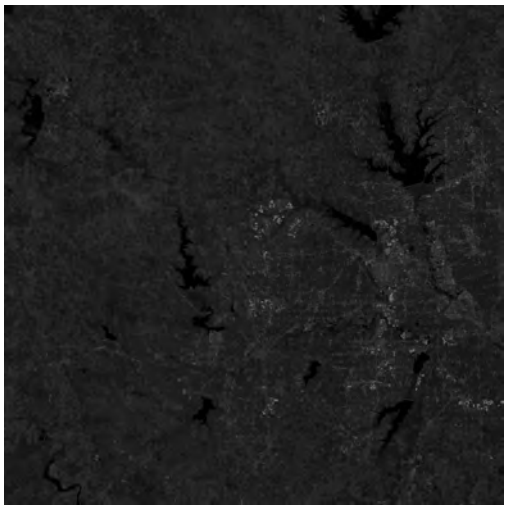


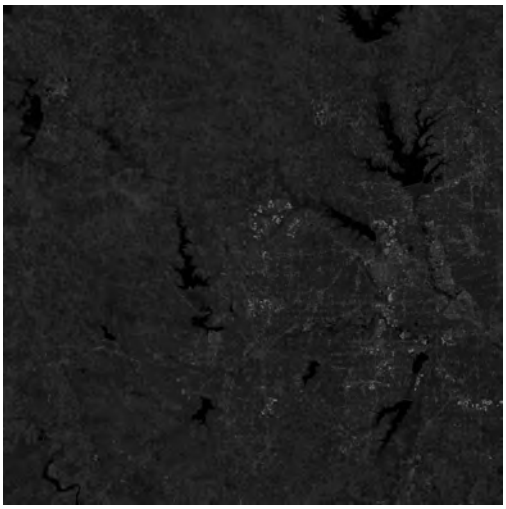
FIGURE A.8. Satellite Sample 2 SSIM Matrix Corruption 40%



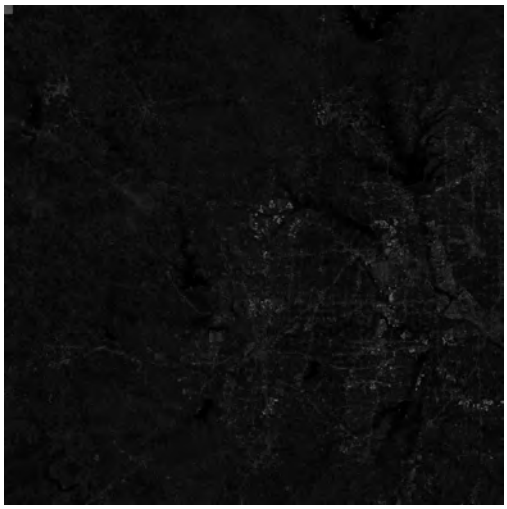
(A) Original L2 Fusion



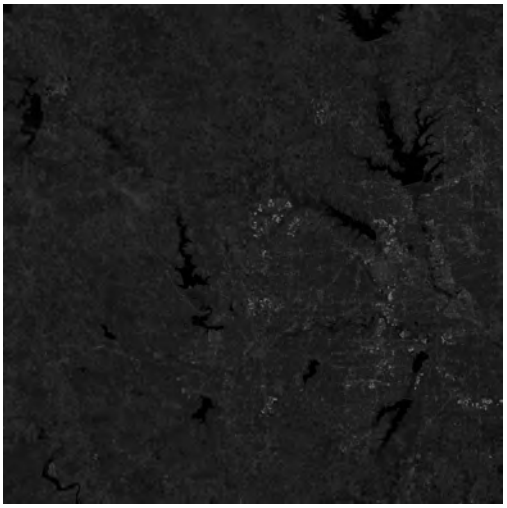
(B) Original L2 Controlled Fusion



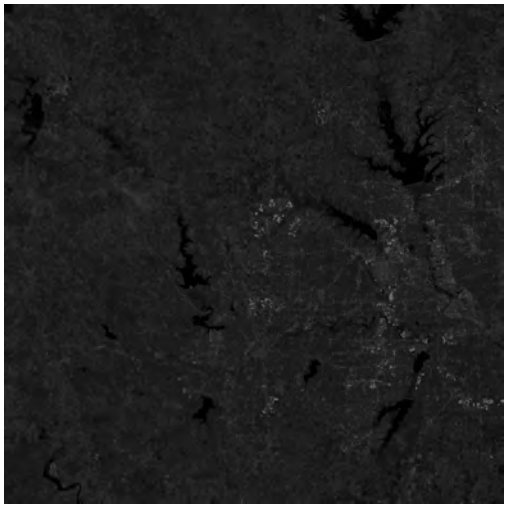
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE A.9. Satellite Sample 2 Images Corruption 50%

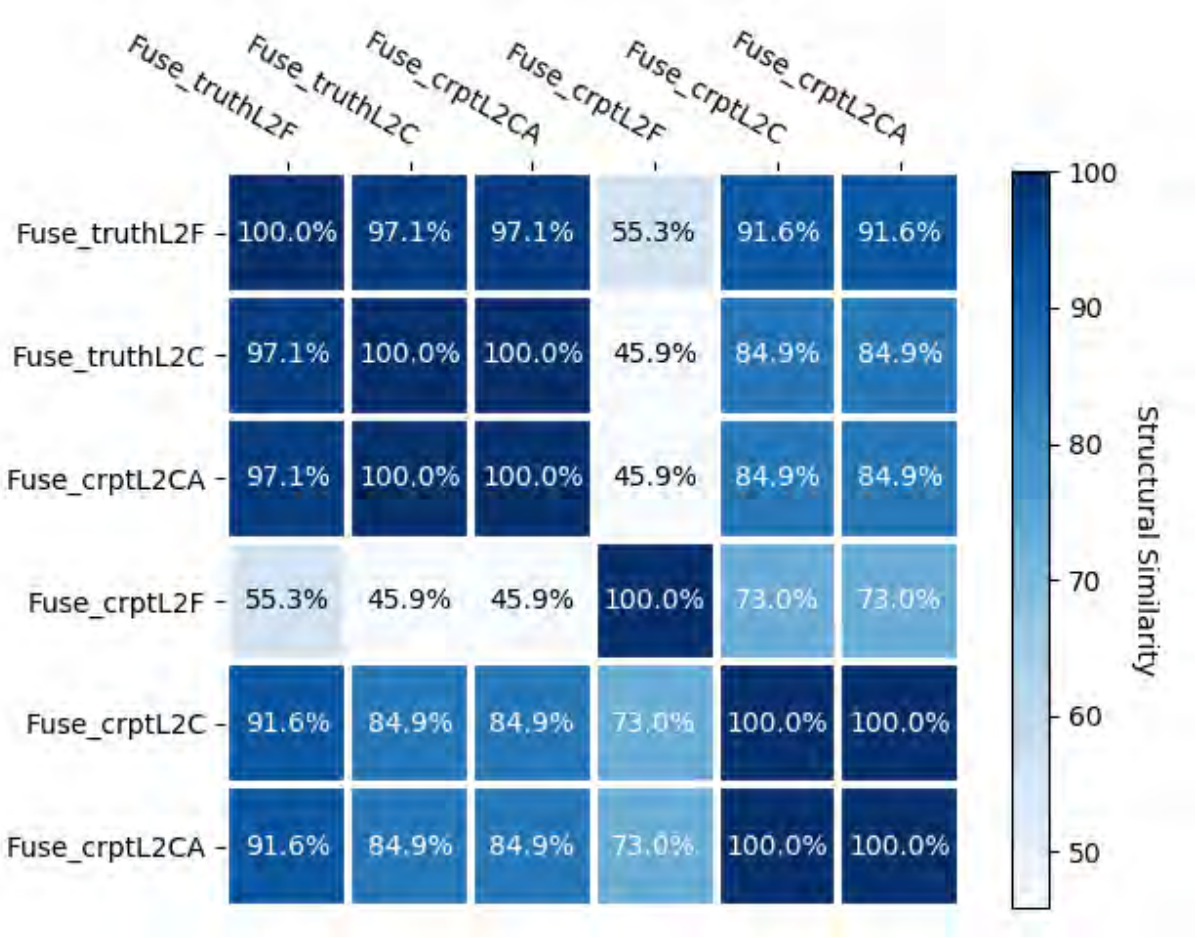
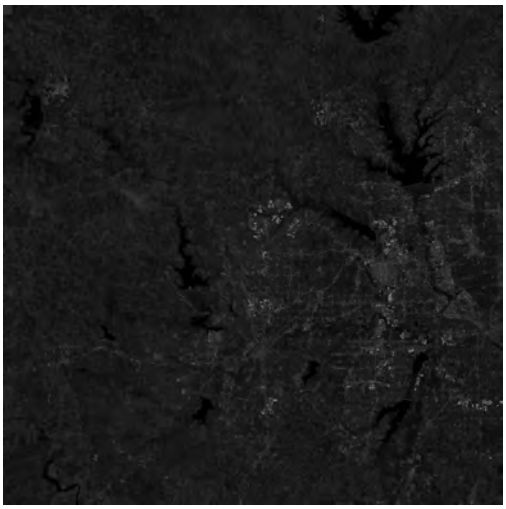
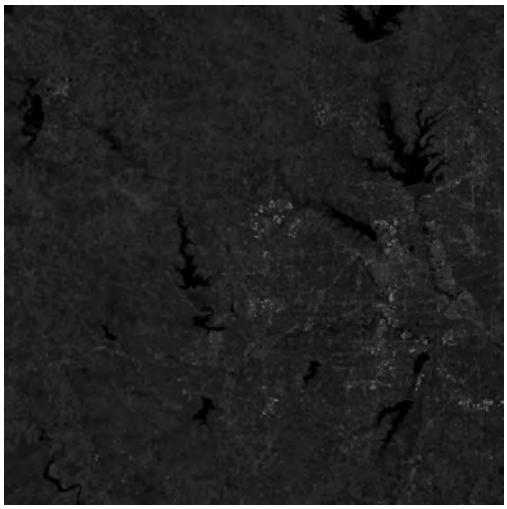


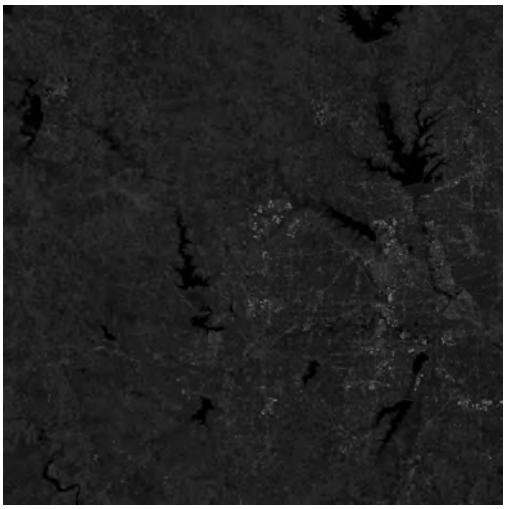
FIGURE A.10. Satellite Sample 2 SSIM Matrix Corruption 50%



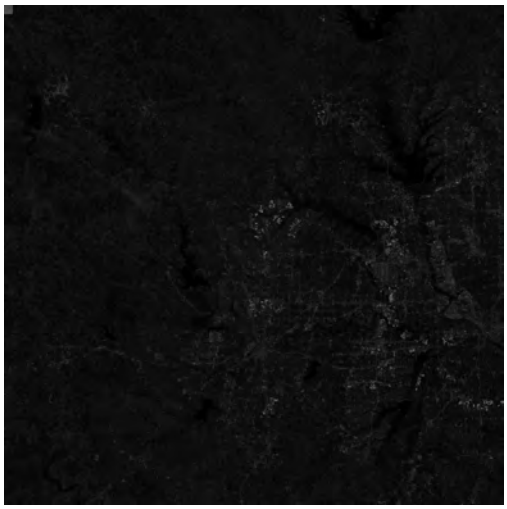
(A) Original L2 Fusion



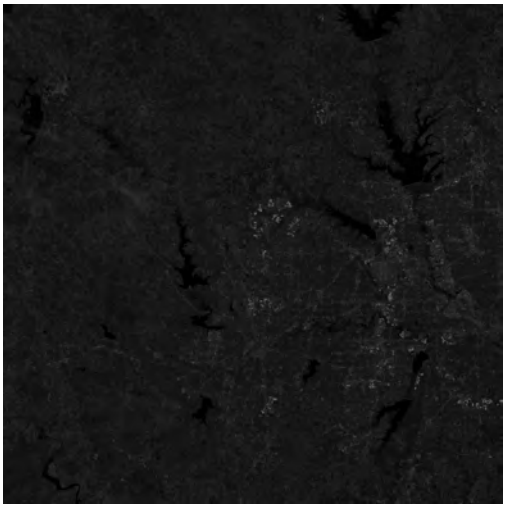
(B) Original L2 Controlled Fusion



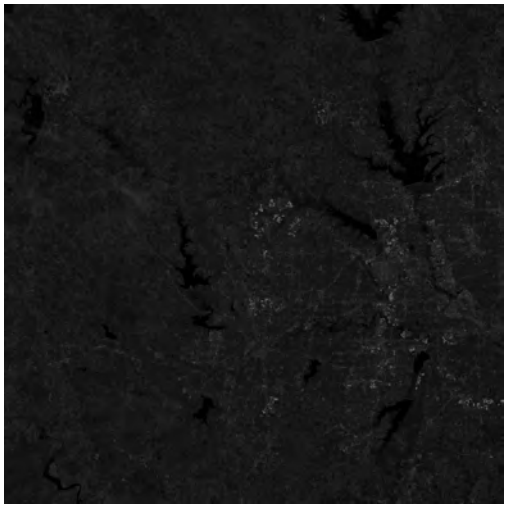
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE A.11. Satellite Sample 2 Images Corruption 55%

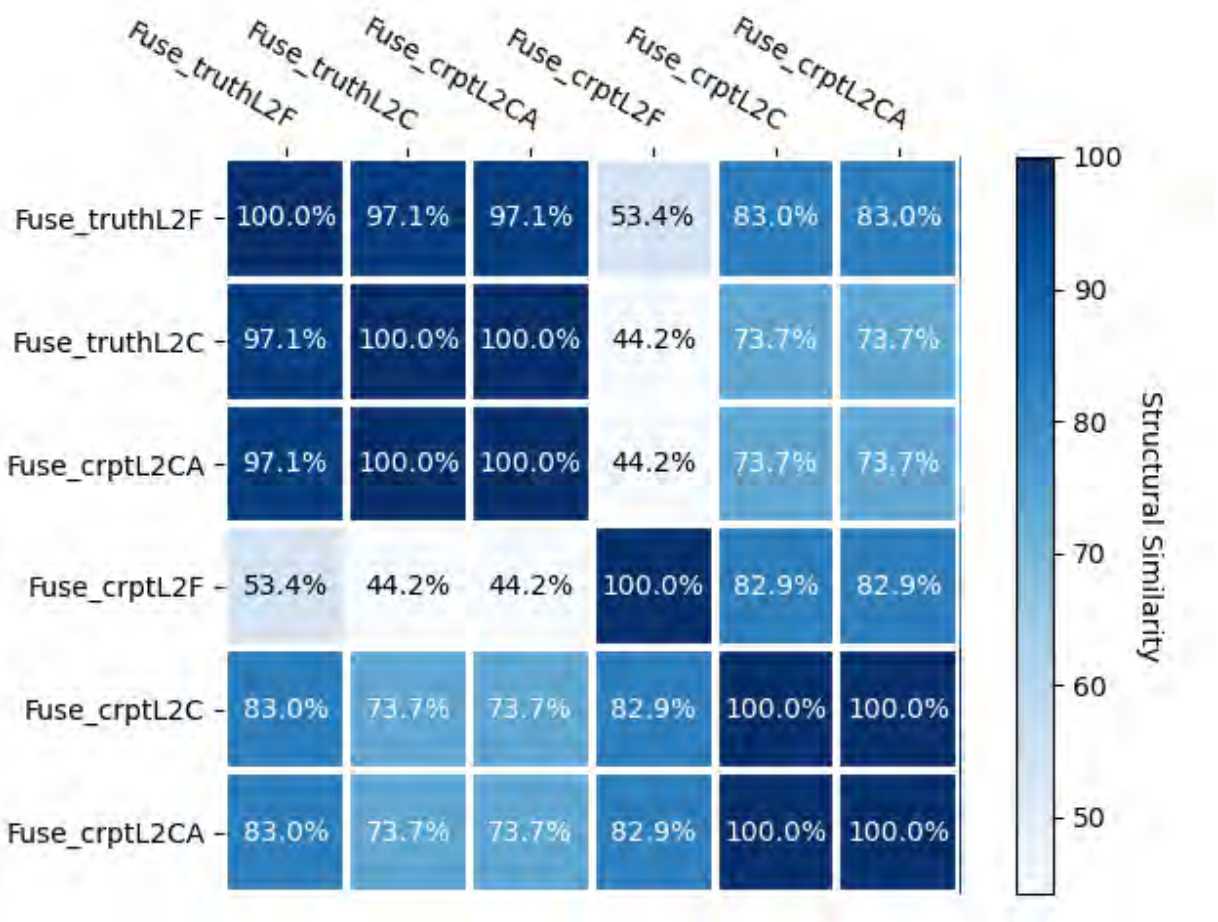
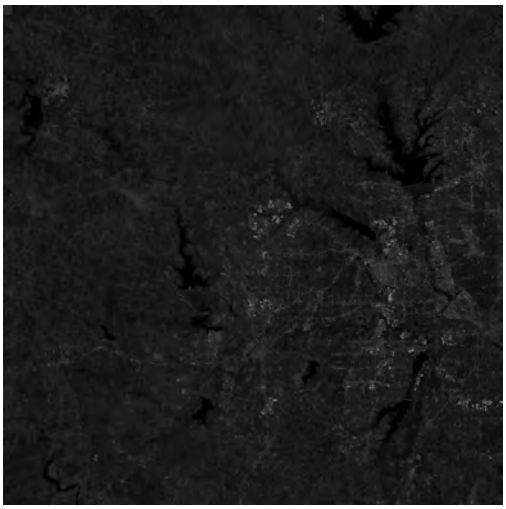
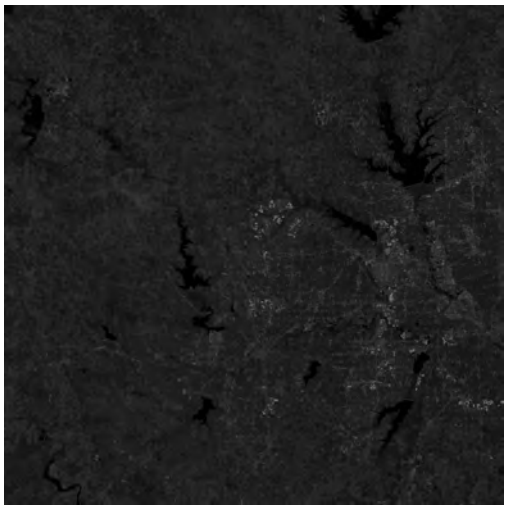


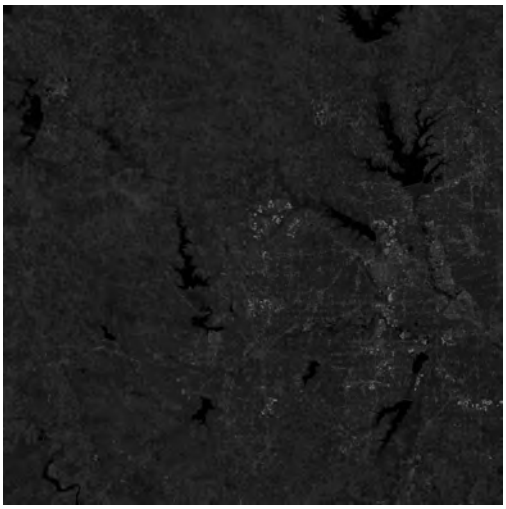
FIGURE A.12. Satellite Sample 2 SSIM Matrix Corruption 55%



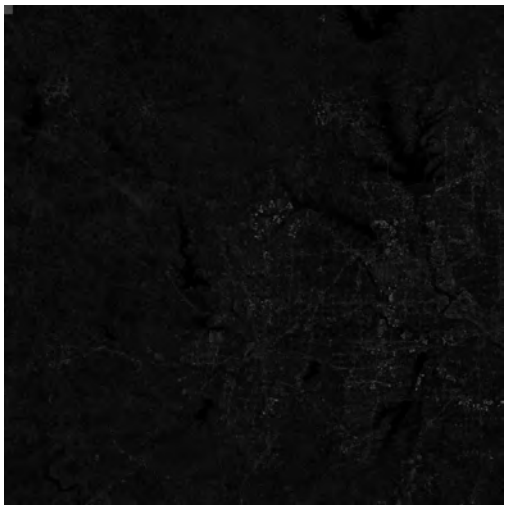
(A) Original L2 Fusion



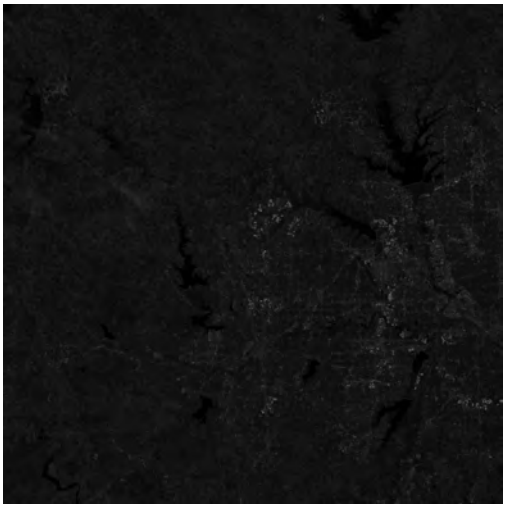
(B) Original L2 Controlled Fusion



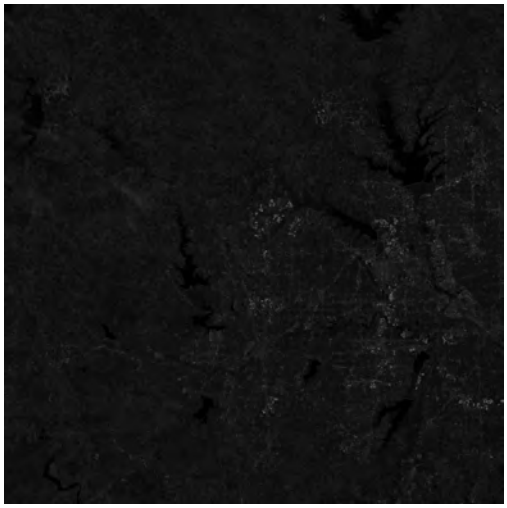
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE A.13. Satellite Sample 2 Images Corruption 70%

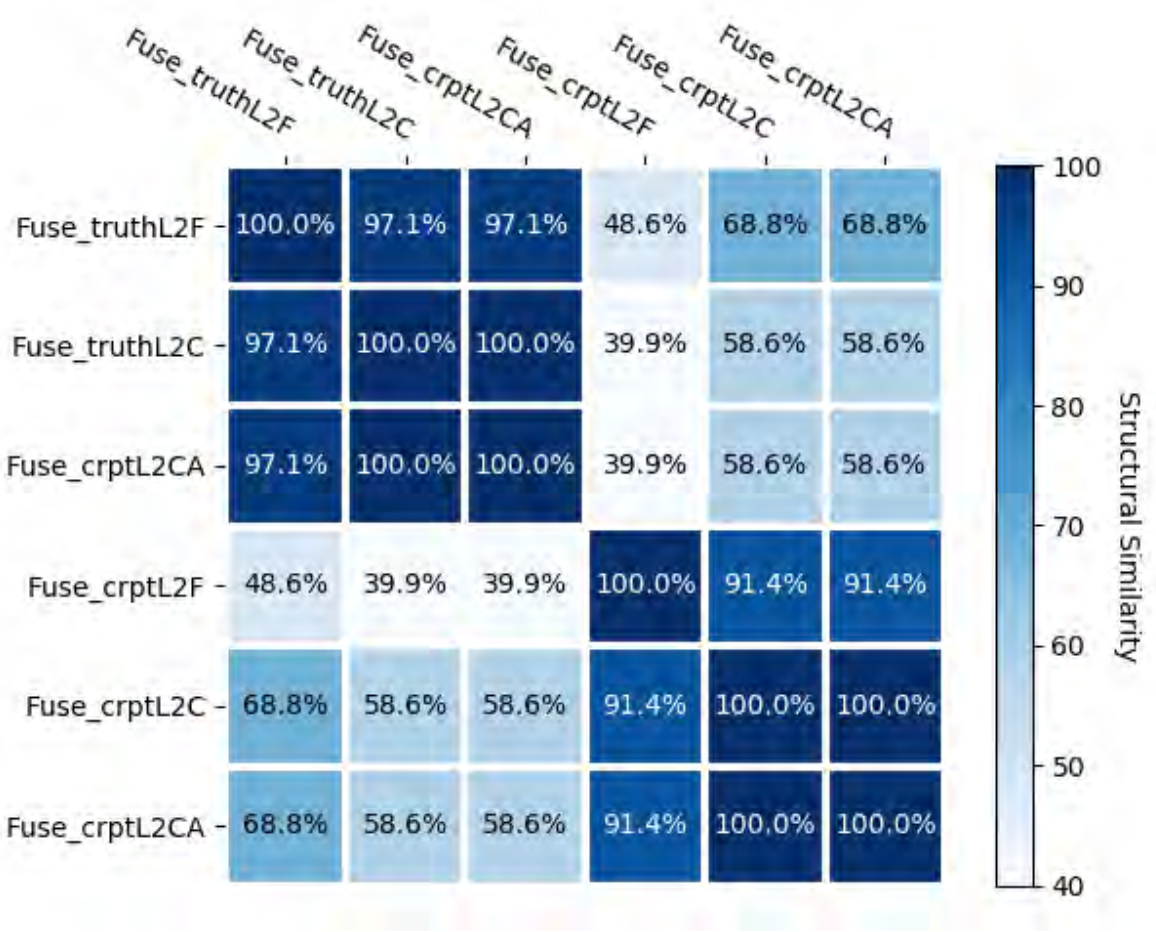
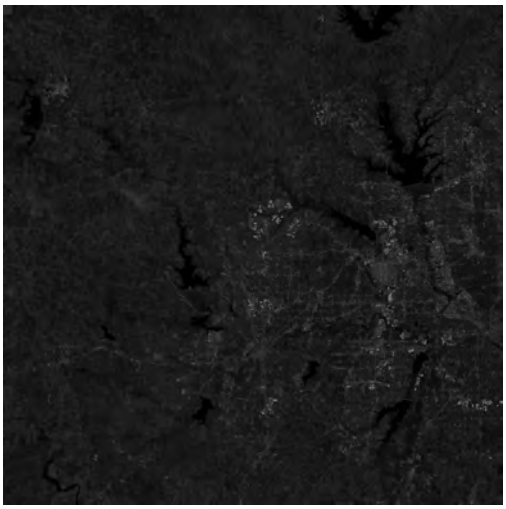
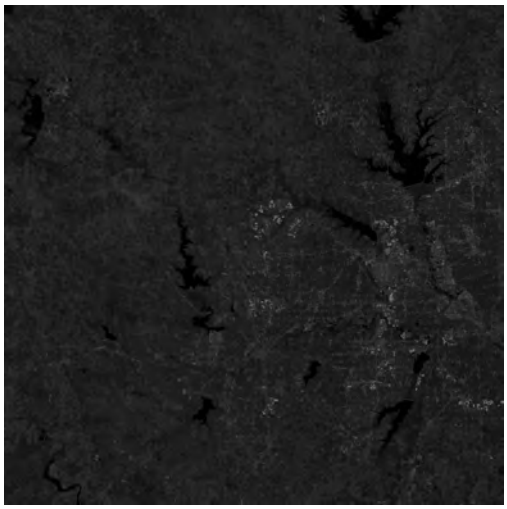


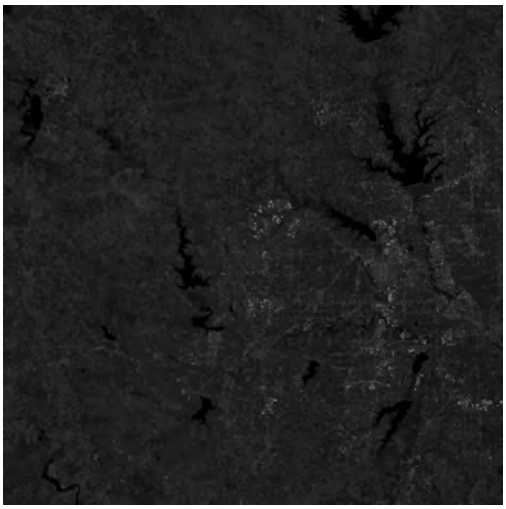
FIGURE A.14. Satellite Sample 2 SSIM Matrix Corruption 70%



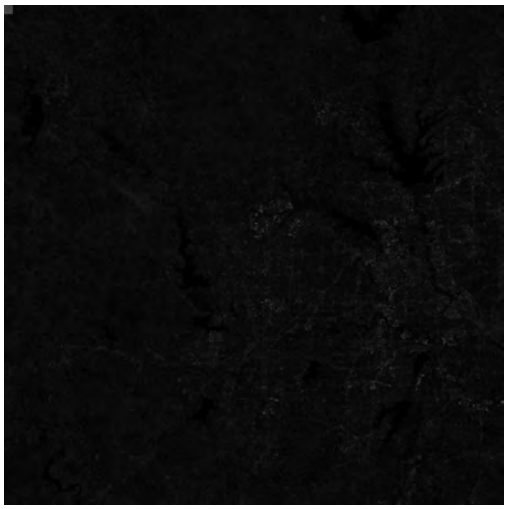
(A) Original L2 Fusion



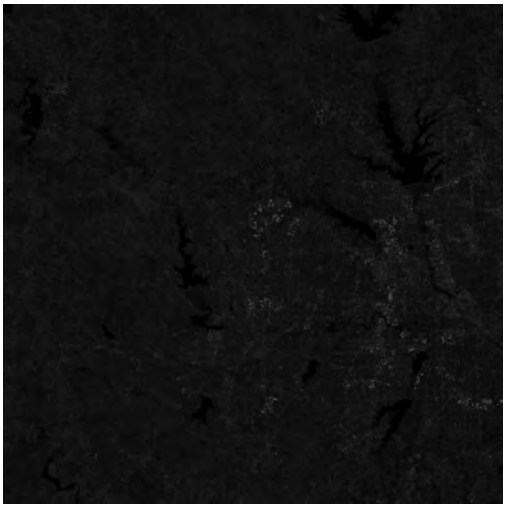
(B) Original L2 Controlled Fusion



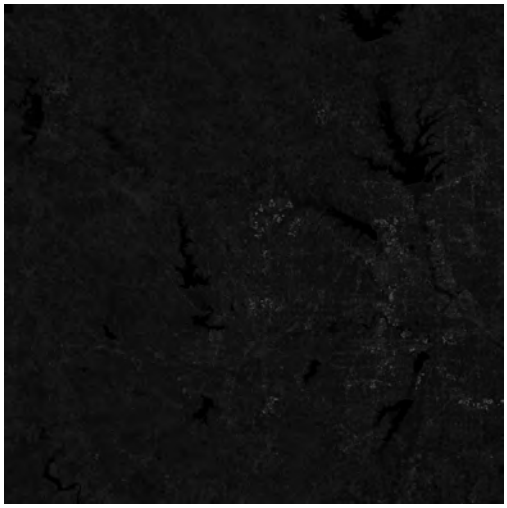
(C) Original L2 Autoencoder Fusion



(D) Corrupted L2 Fusion



(E) Corrupted L2 Controlled Fusion



(F) Corrupted L2 Autoencoder Fusion

FIGURE A.15. Satellite Sample 2 Images Corruption 90%

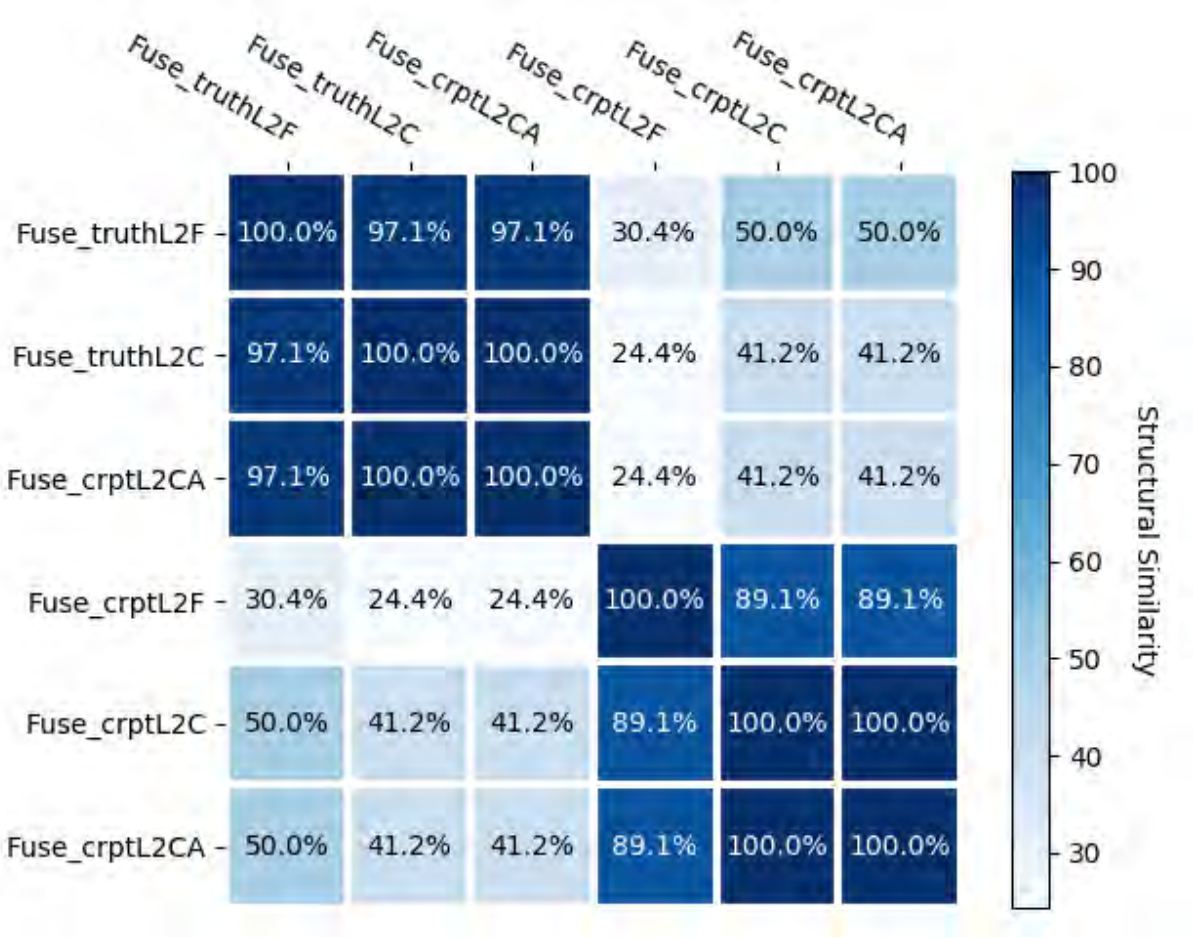


FIGURE A.16. Satellite Sample 2 SSIM Matrix Corruption 90%

REFERENCES

[1] R. Abiko and M. Ikehara, *Blind denoising of mixed gaussian-impulse noise by single cnn*, ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), May 2019, pp. 1717–1721.

[2] B. F. Almubarak, Y. I. Aleisa, B. Liu, and A. H. Muqaibel, *Global pca for in-field compression of seismic data acquisition*, 2017 9th IEEE-GCC Conference and Exhibition (GCCCE), May 2017, pp. 1–4.

[3] A. Azarang, H. E. Manoochehri, and N. Kehtarnavaz, *Convolutional autoencoder-based multispectral image fusion*, IEEE Access 7 (2019), 35673–35683.

[4] C. P. Bailey, S. Chamadia, and D. A. Pados, *An alternative signature design using l1 principal components for spread-spectrum steganography*, ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020, pp. 2693–2696.

[5] R. Bellman, Rand Corporation, and Karreman Mathematics Research Collection, *Dynamic programming*, Rand Corporation research study, Princeton University Press, 1957.

[6] Richard E. Bellman and Stuart E. Dreyfus, *Applied Dynamic Programming*, Princetown University Press, 1962.

[7] Narayan Bhosale, *Analysis of effect of noise removal filters on noisy remote sensing images*, The Journal of Scientific and Engineering Research (2013), 1511–1514.

[8] Irene Biavetti, Sotiris Karetsos, Andrej Ceglar, Andrea Toreti, and Panos Panagos, *European meteorological data: contribution to research, development, and policy support*, Second International Conference on Remote Sensing and Geoinformation of the Environment (RSCy2014) (Diofantos G. Hadjimitsis, Kyriacos Themistocleous, Silas Michaelides, and Giorgos Papadavid, eds.), vol. 9229, International Society for Optics and Photonics, SPIE, 2014, pp. 31 – 39.

[9] E. Brophy, W. Muehlhausen, A. F. Smeaton, and T. E. Ward, *Cnns for heart rate*

estimation and human activity recognition in wrist worn sensing applications, 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2020, pp. 1–6.

[10] J. Canny, *A computational approach to edge detection*, IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-8 (1986), no. 6, 679–698.

[11] Centers for Disease Control and Prevention, *Heart disease facts*, CDC.gov https://www.cdc.gov/heartdisease/facts.htm.

[12] ———, *Underlying cause of death, 1999-2019*, "CDC.gov https://wonder.cdc.gov/controller/datarequest/D76".

[13] S. Chamadia and D. A. Pados, *Optimal sparse l1-norm principal-component analysis*, 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), March 2017, pp. 2686–2690.

[14] Navneet Dalal and Bill Triggs, *Histograms of oriented gradients for human detection*, 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), vol. 1, IEEE, 2005, pp. 886–893.

[15] J. C. Dunn, *Group averaged linear transforms that detect corners and edges*, IEEE Transactions on Computers C-24 (1975), no. 12, 1191–1201.

[16] A. Ekin and R. Jasinschi, *Temporal detection and processing of transparent overlays for video indexing and enhancement*, 2005 13th European Signal Processing Conference, 2005, pp. 1–4.

[17] James Elliott, F. Mueller, M. Stoyanov, and C. Webster, *Quantifying the impact of single bit flips on floating point arithmetic*, 2013.

[18] European Space Agency, *Sentinel data*, https://sentinel.esa.int/web/sentinel/sentinel-data-access, 2020.

[19] Pedro Felzenszwalb, David McAllester, and Deva Ramanan, *A discriminatively trained, multiscale, deformable part model*, 2008 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2008, pp. 1–8.

[20] Karl Pearson F.R.S., *Liii. on lines and planes of closest fit to systems of points in space*,

The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2 (1901), no. 11, 559–572.

[21] G. Golub and W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, Journal of the Society for Industrial and Applied Mathematics Series B Numerical Analysis 2 (1965), no. 2, 205–224.

[22] REINSCH C. GOLUB, G.H., *Handbook series linear algebra. singular value decomposition and least squares solutions.*, Numerische Mathematik 14 (1970), 403–420.

[23] Charles R. Harris, K. Jarrod Millman, St'efan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fern'andez del R'ıo, Mark Wiebe, Pearu Peterson, Pierre G'erard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant, *Array programming with NumPy*, Nature 585 (2020), no. 7825, 357–362.

[24] C. He and J. Jeng, *Feature selection of weather data with interval principal component analysis*, 2016 International Conference on System Science and Engineering (ICSSE), July 2016, pp. 1–4.

[25] Berthold Klaus Paul Horn (ed.), *Robot vision*, MIT Press, Cambridge, MA, USA, 1986.

[26] R. James, A. M. Jolly, C. Anjali, and D. Michael, *Image denoising using adaptive pca and svd*, 2015 Fifth International Conference on Advances in Computing and Communications (ICACC), 2015, pp. 383–386.

[27] Lorenzo E Jaques, Arthur C. Depoian II, Dong Xie, Colleen P. Bailey, and Parthasarathy Guturu, *A machine learning approach to medical data identification through principal component analysis*, Big Data III: Learning, Analytics, and Applications (Fauzia Ahmad, ed.), International Society for Optics and Photonics, SPIE, 2021.

[28] Kinjal A Joshi and Darshak G Thakore, *A survey on moving object detection and tracking in video surveillance system*, International Journal of Soft Computing and Engineering 2 (2012), no. 3, 44–48.

[29] Mohammad Kachuee, Shayan Fazeli, and Majid Sarrafzadeh, *Ecg heartbeat classification: A deep transferable representation*, 2018 IEEE International Conference on Healthcare Informatics (ICHI) (2018).

[30] George N. Karystinos, *Optimal algorithms for binary, sparse, and l1-norm principal component analysis*, pp. 339–382, Springer New York, New York, NY, 2014.

[31] V Kastrinaki, Michalis Zervakis, and Kostas Kalaitzakis, *A survey of video processing techniques for traffic applications*, Image and vision computing 21 (2003), no. 4, 359–381.

[32] Eric King, Arthur C. Depoian II, and Colleen P. Bailey, *Weighted principal component analysis fusion of satellite telemetry data*, Remote Sensing of the Ocean, Sea Ice, Coastal Waters, and Large Water Regions 2020 (Charles R. Bostater Jr., Xavier Neyt, and Françoise Viallefont-Robinet, eds.), vol. 11529, International Society for Optics and Photonics, SPIE, 2020, pp. 9 – 15.

[33] Diederik P. Kingma and Jimmy Ba, *Adam: A method for stochastic optimization*, 2014, cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

[34] Y. Kolokolov and A. Monovskaya, *Estimating of temperature abnormalities in local climate dynamics*, 2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), vol. 2, Sep. 2015, pp. 598–604.

[35] S. Kundu, P. P. Markopoulos, and D. A. Pados, *Fast computation of the l1-principal component of real-valued data*, 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), May 2014, pp. 8028–8032.

[36] C. Lanaras, J. Bioucas-Dias, E. Baltsavias, and K. Schindler, *Super-resolution of multispectral multiresolution images from a single sensor*, 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017, pp. 1505–1513.

[37] Charis Lanaras, José Bioucas-Dias, Silvano Galliani, Emmanuel Baltsavias, and Konrad Schindler, *Super-resolution of sentinel-2 images: Learning a globally applicable deep*

neural network, ISPRS Journal of Photogrammetry and Remote Sensing 146 (2018), 305 – 319.

[38] C. C. Lee, *Elimination of redundant operations for a fast sobel operator*, IEEE Transactions on Systems, Man, and Cybernetics SMC-13 (1983), no. 2, 242–245.

[39] R. C. T. Lee, Y. H. Chin, and S. C. Chang, *Application of principal component analysis to multikey searching*, IEEE Transactions on Software Engineering SE-2 (1976), no. 3, 185–193.

[40] Taiyong Li and Min Zhou, *Ecg classification using wavelet packet entropy and random forests*, Entropy 18 (2016), no. 8.

[41] Z. Li and H. Leung, *Fusion of multispectral and panchromatic images using a restoration-based method*, IEEE Transactions on Geoscience and Remote Sensing 47 (2009), no. 5, 1482–1491.

[42] T. Lili and H. Wei, *Portable ecg monitoring system design*, 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Oct 2019, pp. 1370–1373.

[43] Tony Lindeberg, *Scale invariant feature transform*, (2012).

[44] Z. Liu, W. Q. Yan, and M. L. Yang, *Image denoising based on a cnn model*, 2018 4th International Conference on Control, Automation and Robotics (ICCAR), April 2018, pp. 389–393.

[45] NCAA March Madness, *2018 March Madness NCAA title game: Villanova v. Michigan (FULL)*, Youtube, 10 October 2018 `https://youtu.be/cbe_R73l4Lk`, (Accessed: 15 November 2019).

[46] ———, *Virginia wins 2019 National Championship vs. Texas Tech (FULL GAME)*, Youtube, 14 May 2019 `https://youtu.be/7N9KYZrJpOc`, (Accessed: 15 November 2019).

[47] P. Markopoulos, G. N. Karystinos, and D. Pados, *Some options for l1-subspace signal processing*, ISWCS, 2013.

[48] P. P. Markopoulos, S. Kundu, S. Chamadia, and D. A. Pados, *L1-norm principal-*

component analysis via bit flipping, 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), Dec 2016, pp. 326–332.

[49] ———, *Efficient l1-norm principal-component analysis via bit flipping*, IEEE Transactions on Signal Processing 65 (2017), no. 16, 4252–4264.

[50] P. P. Markopoulos, S. Kundu, and D. A. Pados, *L1-fusion: Robust linear-time image recovery from few severely corrupted copies*, 2015 IEEE International Conference on Image Processing (ICIP), Sep. 2015, pp. 1225–1229.

[51] Panos Markopoulos, Sandipan Kundu, Shubham Chamadia, and Dimitris Pados, *L1-norm principal-component analysis via bit flipping*, 12 2016, pp. 326–332.

[52] Tom Miltonberger and Hans Muller, *A True 2D Edge Detector*, Applications of Artificial Intelligence III (John F. Gilmore, ed.), vol. 0635, International Society for Optics and Photonics, SPIE, 1986, pp. 345 – 351.

[53] S. K. Mohapatra, A. Upadhyay, and C. Gola, *Rainfall prediction based on 100 years of meteorological data*, 2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN), 2017, pp. 162–166.

[54] George B Moody and Roger G Mark, *The impact of the mit-bih arrhythmia database*, IEEE Engineering in Medicine and Biology Magazine 20 (2001), no. 3, 45–50.

[55] B. Moore, *Principal component analysis in linear systems: Controllability, observability, and model reduction*, IEEE Transactions on Automatic Control 26 (1981), no. 1, 17–32.

[56] R. M. Nieto and J. M. M. Sánchez, *An automatic system for sports analytics in multi-camera tennis videos*, 2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance, Aug 2013, pp. 438–442.

[57] Shipra Ojha and Sachin Sakhare, *Image processing techniques for object tracking in video surveillance-a survey*, 2015 International Conference on Pervasive Computing (ICPC), IEEE, 2015, pp. 1–6.

[58] Ryosuke Okuta, Yuya Unno, Daisuke Nishino, Shohei Hido, and Crissman Loomis, *Cupy: A numpy-compatible library for nvidia gpu calculations*, Proceedings of Workshop

on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS), 2017.

[59] E. E. C. Osorio, SeokYoon Kang, Bum-Su Kim, JoHo Lim, Kyong Hoon Kim, and Ki-Il Kim, *Development of data collecting system for forecasting with meteorological sensors*, 2017 International Conference on Information Networking (ICOIN), Jan 2017, pp. 453–456.

[60] Nobuyuki Otsu, *A threshold selection method from gray-level histograms*, IEEE transactions on systems, man, and cybernetics 9 (1979), no. 1, 62–66.

[61] Frosti Palsson, Johannes R. Sveinsson, and Magnus O. Ulfarsson, *Sentinel-2 image fusion using a deep residual network*, Aug 2018.

[62] Vaibhav Pandit and R. Bhiwani, *Image fusion in remote sensing applications: A review*, International Journal of Computer Applications 120 (2015), 22–32.

[63] Himani S Parekh, Darshak G Thakore, and Udesang K Jaliya, *A survey on object detection and tracking methods*, International Journal of Innovative Research in Computer and Communication Engineering 2 (2014), no. 2, 2970–2978.

[64] J. H. Park, J. Hyeon Kim, and S. I. Cho, *The analysis of cnn structure for image denoising*, 2018 International SoC Design Conference (ISOCC), Nov 2018, pp. 220–221.

[65] S. Parthasarathy and C. C. Aggarwal, *On the use of conceptual reconstruction for mining massively incomplete data sets*, IEEE Transactions on Knowledge and Data Engineering 15 (2003), no. 6, 1512–1521.

[66] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala, *Pytorch: An imperative style, high-performance deep learning library*, Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035.

[67] U. Patil and U. Mudengudi, *Image fusion using hierarchical pca.*, 2011 International Conference on Image Information Processing, Nov 2011, pp. 1–6.

[68] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python*, Journal of Machine Learning Research 12 (2011), 2825–2830.

[69] John G. Proakis and Dimitris K. Manolakis, *Digital signal processing (4th edition)*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006.

[70] A. M. Rateb, *A fast compressed sensing decoding technique for remote ecg monitoring systems*, IEEE Access 8 (2020), 197124–197133.

[71] H. R. Shahdoosti and H. Ghassemian, *Spatial pca as a new method for image fusion*, The 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP 2012), May 2012, pp. 090–094.

[72] Manish Sharma, R. Tan, and U. Acharya, *Automated heartbeat classification and detection of arrhythmia using optimal orthogonal wavelet filters*, Informatics in Medicine Unlocked 16 (2019), 100221.

[73] Irwin Sobel, *An isotropic 3x3 image gradient operator*, Presentation at Stanford A.I. Project 1968 (2014).

[74] M. Stein, T. Breitkreutz, J. Haussler, D. Seebacher, C. Niederberger, T. Schreck, M. Grossniklaus, D. Keim, and H. Janetzko, *Revealing the invisible: Visual analytics and explanatory storytelling for advanced team sport analysis*, 2018 International Symposium on Big Data Visual and Immersive Analytics (BDVA), Oct 2018, pp. 1–9.

[75] George Stockman and Linda G. Shapiro, *Computer vision*, 1st ed., Prentice Hall PTR, USA, 2001.

[76] Y. Su, J. Li, A. Plaza, A. Marinoni, P. Gamba, and Y. Huang, *Deep auto-encoder network for hyperspectral image unmixing*, IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium, July 2018, pp. 6400–6403.

[77] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders, *Selective search for object recognition*, International journal of computer vision 104 (2013), no. 2, 154–171.

[78] A. Van Rotterdam, *Limitations and difficulties in signal processing by means of the principal-components analysis*, IEEE Transactions on Biomedical Engineering BME-17 (1970), no. 3, 268–269.

[79] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors, *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python*, Nature Methods 17 (2020), 261–272.

[80] Zhikang Xiao, Yang Zou, and Zhen Wang, *An improved dynamic double threshold Canny edge detection algorithm*, MIPPR 2019: Pattern Recognition and Computer Vision (Nong Sang, Jayaram K. Udupa, Yuehuan Wang, and Zhenbing Liu, eds.), vol. 11430, International Society for Optics and Photonics, SPIE, 2020, pp. 234 – 241.

[81] K. Xie, X. Ning, X. Wang, D. Xie, J. Cao, G. Xie, and J. Wen, *Recover corrupted data in sensor networks: A matrix completion solution*, IEEE Transactions on Mobile Computing 16 (2017), no. 5, 1434–1448.

[82] Junjie Yan, Zhen Lei, Longyin Wen, and Stan Z Li, *The fastest deformable part model for object detection*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 2497–2504.

[83] Wenkao Yang, Jing Wang, and Jing Guo, *A novel algorithm for satellite images fusion based on compressed sensing and pca*, Mathematical Problems in Engineering 2013 (2013).

[84] Fumito Yoshikawa, Kazuo Toraichi, N Wada, Nobuyuki Otsu, Hiroyasu Nakai, Mitsuru Mitsumoto, Kazuki Katagishi, and Kwan Paul Wing Hing, *Feature extraction algorithm for beef marbling*, 1999 IEEE Pacific Rim Conference on Communications, Computers

and Signal Processing (PACRIM 1999). Conference Proceedings (Cat. No. 99CH36368), IEEE, 1999, pp. 209–212.

[85] X. Zhang and Y. Lian, *A 300-mv 220-nw event-driven adc with real-time qrs detection for wearable ecg sensors*, IEEE Transactions on Biomedical Circuits and Systems 8 (2014), no. 6, 834–843.

[86] Linchao Zhu, Zhongwen Xu, Yi Yang, and Alexander G Hauptmann, *Uncovering the temporal context for video question answering*, International Journal of Computer Vision 124 (2017), no. 3, 409–421.

[87] C Lawrence Zitnick and Piotr Dollár, *Edge boxes: Locating object proposals from edges*, European conference on computer vision, Springer, 2014, pp. 391–405.